PENGEMBANGAN SISTEM INFORMASI MANAJEMEN PENGASUHAN SANTRI DI PESANTREN (STUDI PADA: TAZKIA *INTERNATIONAL ISLAMIC BOARDING SCHOOL* (IIBS))

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Wahyu Cahya Fibrianto
NIM: 145150400111016

PROGRAM STUDI SISTEM INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2021

PENGESAHAN

PENGEMBANGAN SISTEM INFORMASI MANAJEMEN PENGASUHAN SANTRI DI PESANTREN (STUDI PADA: TAZKIA *INTERNATIONAL ISLAMIC BOARDING SCHOOL* (IIBS))

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Wahyu Cahya Fibrianto
NIM: 145150400111016

Skripsi ini telah diuji dan dinyatakan lulus pada
9 Juli 2021
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Satrio Agung Wicaksono, S.Kom., M.Kom.	Fajar Pradana, S.ST., M.Eng.
NIP. 198605212012121001	NIP. 198711212015041004

Mengetahui
Ketua Jurusan **Sistem Informasi**

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS BRAWIJAYA FAKULTAS ILMU KOMPUTER JURUSAN SISTEM INFORMASI

Issa Arwani, S.Kom., M.Sc.
NIP. 198309222012121003

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 22 Juni 2021

Wahyu Cahya Fibrianto

NIM: 145150400111016

PRAKATA

Puji dan syukur penulis panjatkan kehadirat Allah SWT, karena berkat rahmat dan karunia-Nyalah penulis dapat menyelesaikan skripsi yang berjudul "PENGEMBANGAN SISTEM INFORMASI MANAJEMEN PENGASUHAN SANTRI DI PESANTREN (STUDI PADA: TAZKIA *INTERNATIONAL ISLAMIC BOARDING SCHOOL* (IIBS))". Adapun maksud dan tujuan dari penulisan skripsi ini adalah untuk Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer. Selama penelitian dan penulisan skripsi ini banyak sekali hambatan yang penulis alami, namun berkat bantuan, dorongan serta bimbingan dari berbagai pihak, akhirnya skripsi ini dapat terselesaikan dengan baik. Penulis beranggapan bahwa skripsi ini merupakan karya terbaik yang dapat penulis persembahkan. Tetapi penulis menyadari bahwa tidak tertutup kemungkinan didalamnya terdapat kekurangan-kekurangan. Oleh karena itu kritik dan saran yang membangun sangat penulis harapkan. Akhir kata, semoga skripsi ini dapat bermanfaat bagi penulis khususnya dan bagi parapembaca pada umumnya.

Malang, 22 Juni 2021

Penulis

ABSTRAK

Wahyu Cahya Fibrianto, PENGEMBANGAN SISTEM INFORMASI MANAJEMEN PENGASUHAN SANTRI DI PESANTREN (STUDI PADA: TAZKIA *INTERNATIONAL ISLAMIC BOARDING SCHOOL* (IIBS))

Pembimbing: Satrio Agung Wicaksono, S.Kom., M.Kom. dan Fajar Pradana, S.ST., M.Eng.

Penelitian ini bertujuan untuk mengetahui tingkat penerimaan pengguna terhadap kecepatan sistem informasi manajemen kepengasuhan dalam proses pencatatan dan pelaporan serta efektivitas proses penilaian aktivitas kepengasuhan santri pada pondok pesantren Tazkia Internasional Islamic Boarding School (IIBS) menggunakan sistem informasi manajemen pengasuhan. Metode pengembangan sistem informasi menggunakan *Rational Unified Process* (RUP). Aktivitas kepengasuhan memiliki tiga proses bisnis, antara lain pengelolaan kamar, penilaian kemandirian dan penilaian peribadatan. Tahap pemodelan proses bisnis didapatkan model proses bisnis to-be yang digunakan sebagai dasar analisis kebutuhan dan perancangan sistem informasi. Implementasi sistem menggunakan pola arsitektur *Model-View-Controll*er (MVC) dengan memanfaatkan kerangka kerja Codeigniter (CI). Pengujian validasi dilakukan untuk mengetahui sistem yang dibangun sudah sesuai dengan analisis kebutuhan. Hasil pengujian validasi menunjukkan sistem yang dibangun 100% valid. Dari analisis pengujian *user acceptance testing* didapatkan hasil bahwa pengguna sangat setuju bahwa sistem yang dikembangkan sesuai dengan apa yang dibutuhkan dengan skor sebesar 80%. Pengguna juga sangat setuju bahwa sistem mudah digunakan dengan skor 81,7%. Selain itu pengguna sangat setuju bahwa sistem mempercepat proses penilaian dan pelaporan kemandirian dan peribadatan sebesar 87,5%. Terakhir, pengguna juga sangat setuju sistem membantu proses kegiatan kemandirian dan peribadatan kapanpun dan dimanapun sebesar 85%.

Kata kunci: *Rational Unified Process (RUP),* Tazkia IIBS*,* Pesantren*, User Acceptance Testing (UAT)*

ABSTRACT

Wahyu Cahya Fibrianto, *THE DEVELOPMENT OF STUDENT CARE MANAGEMENT INFORMATION SYSTEMS IN ISLAMIC BOARDING SCHOOL (STUDY AT: TAZKIA INTERNATIONAL ISLAMIC BOARDING SCHOOL (IIBS))*

Supervisors: Satrio Agung Wicaksono, S.Kom., M.Kom. and Fajar Pradana, S.ST., M.Eng.

This study aims to determine the level of user acceptance of the speed of the parenting management information system in the recording and reporting process as well as the effectiveness of the process of evaluating student activities at the Tazkia International Islamic Boarding School (IIBS) boarding school using a management information system. The information system development method uses the Rational Unified Process (RUP). Parenting activities have three business processes, including dormitory management, self-reliance assessment and worship assessment. In the business process modeling stage, a to-be business process model is obtained which is used as the basis for needs analysis and information system design. The system implementation uses the Model-View-Controller (MVC) architectural pattern by utilizing the Codeigniter (CI) framework. Validation testing is carried out to find out the system built is in accordance with the needs analysis. The results of the validation test show that the system built is 100% valid. From the analysis of user acceptance testing, it was found that users strongly agree that the system developed is in accordance with what is needed with a score of 80%. Users also strongly agree that the system is easy to use with a score of 81.7%. In addition, users strongly agree that the system accelerates the process of assessing and reporting self-reliance and worship by 87.5%. Finally, users also strongly agree that the system helps the process of self-reliance and worship activities anytime and anywhere by 85%.

Keywords: *Rational Unified Process (RUP), Tazkia IIBS, Boarding School, User Acceptance Testing (UAT)*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Lembaga pendidikan yang berbasis pondok pesantren di Indonesia sudah dikenal lama. Sepanjang berdirinya pondok pesantren, terdapat banyak perubahan pada sistem pendidikan dan manajemennya. Menurut Nizar. S (2007) terdapat tiga macam pesantren dibedakan dari coraknya. Pertama pesantren yang nilai-nilai tradisionalnya masih dipertahankan. Kedua pesantren yang sudah mulai mengadopsi sistem pendidikan modern namun masih memiliki corak tradisional. Terakhir yaitu pesantren yang telah mengalami perubahan secara signifikan, baik sistem unsur-unsur kelembagaannya maupun pendidikannya atau disebut pesantren modern.

Tazkia *International Islamic Boarding School* (IIBS) merupakan lembaga pendidikan berbasis pesantren modern yang menyelenggarakan sistem pendidikan secara menyeluruh dan berimbang dalam ilmu agama dan ilmu umumnya. Untuk mewujudkan sistem pendidikan tersebut, Tazkia IIBS membagi unsur kelembagaannya menjadi dua unit utama yaitu akademik dan kepesantrenan. Unit akademik merupakan unit yang mengatur urusan proses akademik santri seperti kurikulum akademik, perangkat pembelajaran dan sebagainya. Kedua, unit kepesantrenan yang mengatur tentang urusan pesantren seperti kurikulum diniyah (keagamaan), dakwah dan kepengasuhan. Setiap unit memiliki perannya masing-masing dalam menyelenggarakan sistem pendidikan bagi santri.

Sistem pendidikan di Tazkia IIBS dilakukan pada saat jam sekolah dan luar sekolah. Pembelajaran santri di waktu sekolah dilakukan bersama guru. Ketika di luar waktu sekolah, proses pendidikan dilakukan bersama pengasuh. Sama halnya dengan orang tua, pengasuh memiliki peran penting dalam mendidik santrinya karena sebagian besar waktu santri lebih banyak dihabiskan bersama pengasuh. Pengasuh berkewajiban memberikan pembelajaran, pendampingan, pengawasan dan penilaian. Proses tersebut terus berlangsung dalam aktivitas santri sehari-hari seperti kegiatan ibadah, kemandirian, membaca dan tahfidz Al-Quran.

Berdasarkan hasil wawancara dengan Kepala Kepesantrenan Tazkia IIBS, setiap pengasuh (murabbi) bertanggung jawab atas kurang lebih dua puluh empat santri dan mencatat nilai keseharian setiap santri yang diasuhnya. Pencatatan dilakukan menggunakan lembar penilaian yang dipegang oleh setiap pengasuh. Banyaknya elemen penilaian pada setiap aktivitas dan jumlah santri ditambah pencatatan dilakukan secara manual membuat waktu dari pengasuh banyak dihabiskan untuk proses tersebut. Setiap akhir semester, data dari pengasuh diserahkan ke Kepala Kepesantrenan untuk diolah. Proses penyerahan data dari pengasuh ke Kepala Kepesantrenan sering kali mengalami keterlambatan sehingga menghambat proses pengolahan data selanjutnya. Selain itu, pengasuh maupun Kepala Kepesantrenan tidak dapat mengakses informasi laporan nilai santri sewaktu-waktu karena data dari lembar kertas perlu dilakukan rekapitulasi terlebih dahulu.

Berdasarkan kondisi yang dijelaskan sebelumnya, maka diperlukan adanya sistem informasi yang mampu meningkatkan efisiensi proses pencatatan dan memudahkan aksesibilitas nilai kepengasuhan santri.

Penelitian yang dilakukan Hendri dkk (2015) menunjukkan dengan adanya sistem informasi pendidik dan tenaga kependidikan (PTK), maka pihak manajemen sekolah dimudahkan dalam pengelolaan data transaksional PTK dan mendapatkan informasi yang dibutuhkan manajemen sekolah. Penelitian lain oleh Erliyah dkk (2018) menyimpulkan bahwa untuk rekapitulasi data santri, mengatur jadwal kegiatan dan mencatat data kehadiran menggunakan sistem informasi manajemen kegiatan santri. Selain yang sudah dibahas, sistem informasi juga dapat mendukung pengelolaan data santri, pelanggaran, kegiatan, sanksi, dan kehadiran siswa pada kegiatan tertentu.

Berdasarkan uraian di atas, maka perlu dilakukan pengembangan sistem informasi manajemen pengasuhan santri di Tazkia IIBS. Oleh karena itu, penelitian ini mengangkat judul "Pengembangan Sistem Informasi Manajemen Pengasuhan Santri di Pesantren (Studi Pada: Tazkia *International Islamic Boarding School* (IIBS))". Pengembangan sistem informasi diharapkan mampu menyelesaikan permasalahan terkait efisiensi waktu pencatatan dan memudahkan aksesibilitas nilai santri.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah tersebut, maka rumusan masalah yang akan dikaji sebagai berikut:

1. Bagaimana tingkat penerimaan pengguna terhadap kecepatan sistem informasi pengasuhan santri dalam proses pencatatan nilai santri sampai menghasilkan laporan?
2. Bagaimana tingkat penerimaan pengguna terhadap kecepatan sistem informasi pengasuhan santri dalam pencarian laporan nilai santri?
3. Bagaimana efektivitas proses penilaian santri dengan menggunakan sistem informasi kepengasuhan?

1.3 Tujuan

Tujuan dari penelitian ini adalah sebagai berikut:

1. Mengetahui tingkat penerimaan pengguna terhadap kecepatan sistem informasi pengasuhan santri dalam proses pencatatan nilai santri
2. Mengetahui tingkat penerimaan pengguna terhadap kecepatan sistem informasi pengasuhan santri dalam pencarian laporan nilai santri
3. Mengetahui efektivitas proses penilaian santri menggunakan sistem informasi kepengasuhan

1.4 Manfaat Penelitian

Penelitian yang dilakukan diharapkan dapat memberikan manfaat sebagai berikut:

1. Bagi pengasuh, hasil penelitian ini diharapkan mampu membantu dalam mempercepat pencatatan nilai santri.
2. Bagi pimpinan pesantren, penerapan sistem ini diharapkan dapat membantu dalam aksesibilitas informasi nilai santri secara tepat waktu dan akurat.
3. Bagi peneliti, manfaat yang diharapkan dari seluruh rangkaian penelitian yang dilakukan dapat memperluas wawasan serta menjadi ilmu yang bermanfaat.

1.5 Batasan Masalah

Batasan pada penelitian ini adalah sebagai berikut.

1. Pengelolaan nilai kegiatan santri dilakukan oleh pengasuh dan kepala kepesantrenan Tazkia IIBS.
2. Aktivitas kepengasuhan yang dijadikan objek penelitian adalah kegiatan peribadatan dan kemandirian.
3. Laporan akhir berupa status ketuntasan setiap santri dari aktivitas kemandirian dan peribadatan

1.6 Sistematika Pembahasan

Sistematika dalam penyusunan skripsi ini adalah sebagai berikut:

BAB 1 : PENDAHULUAN

Pada bab ini dipaparkan hasil pendahuluan terkait (1) latar belakang, (2) rumusan masalah, (3) tujuan penelitian, (4) manfaat penelitian, (5) batasan masalah dan (6) sistematika pembahasan.

BAB 2 : LANDASAN KEPUSTAKAAN

Bab ini menyajikan hasil literatur tentang profil Tazkia IIBS, sistem informasi, *Rational Unified Process* (RUP), pemodelan proses bisnis, analisis kebutuhan, desain sistem, implementasi dan pengujian sistem. Dasar literatur dirangkum dari berbagai sumber referensi seperti buku, jurnal dan internet.

BAB 3 : METODOLOGI PENELITIAN

Pada bab ini dijelaskan langkah – langkah yang dilakukan atau metode penelitian yang digunakan dalam melakukan penelitian ini. Metode tersebut memuat segala kegiatan dari awal sampai akhir penelitian. Setiap langkah metode dijelaskan tujuan, proses, dan hasil dari metode yang dilakukan.

BAB 4 : HASIL DAN PEMBAHASAN

Bab ini akan membahas penelitian yang dilakukan. Mulai dari pemodelan proses bisnis, analisis kebutuhan, perancangan, implementasi beserta pengujian. Setiap tahapan dibahas berdasarkan hasil yang sudah didapat.

BAB 5 : PENUTUP

Bab ini menjelaskan kesimpulan dari penelitian yang menjawab rumusan masalah penelitian. Serta memberikan saran untuk penelitian selanjutnya berdasarkan hasil penelitian yang telah dilakukan. Kesimpulan dari penelitian berupa jawaban dari rumusan masalah yang diangkat dalam penelitian ini. Kemudian saran dari penelitian ini dimaksudkan untuk melakukan perbaikan terhadap penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Tinjauan Pustaka

Sebuah penelitian yang telah dilakukan oleh Hendri dkk (2015) yang berjudul "Pengembangan Sistem Informasi Manajemen Pendidik dan Tenaga Kependidikan" menjelaskan tentang penelitian yang dilakukan di SMKN 8 Malang. Penelitian tersebut menjelaskan bagaimana peneliti mengembangkan sistem informasi yang digunakan untuk menyimpan data induk pendidik dan tenaga kependidikan (PTK), presensi kehadiran PTK, dan jurnal mengajar. Pengembangan sistem menggunakan pendekatan *bottom-up* (dari kebutuhan pengguna akhir hingga manajemen). Dari pengamatan tersebut, dapat dikatakan bahwa stabilitas saat menggunakan aplikasi tergolong tinggi. Hal ini terlihat dari catatan *database* yang semakin lengkap dari waktu ke waktu, jumlah data yang terjadi, dan konsistensi antar transaksi operasional. Pengguna akhir tidak hanya merasa bahwa kebutuhan mereka terpenuhi secara efisien dan efektif, tetapi juga menciptakan suasana yang jelas dan nyaman dalam organisasi.

Menganalisis kebutuhan pengguna merupakan langkah penting dalam pengembangan perangkat lunak. Analisis kebutuhan pengguna diakui sebagai tugas penting, karena banyak kegagalan perangkat lunak berasal dari spesifikasi persyaratan yang tidak konsisten, tidak lengkap, atau salah (Wahono, 2003). Dalam penelitian yang dilakukan Wahono (2003), hasil tahap rekayasa persyaratan didokumentasikan dalam spesifikasi kebutuhan yang mencerminkan masalah yang akan dipecahkan antara analisis dan pengguna serta sebagai titik awal untuk fase berikutnya yaitu desain. Penelitian ini digunakan sebagai referensi dalam melakukan analisis kebutuhan pengguna sistem informasi manajemen pengasuhan santri.

Anwar (2014) mengulas kerangka pengembangan perangkat lunak *Rational Unified Process (RUP).* Industri pengembangan perangkat lunak semakin banyak menggunakan RUP. Hal ini berdasarkan pada praktik yang telah diuji berkali-kali di banyak proyek dan area yang berbeda dan, yang terpenting, telah terbukti berhasil. Dengan berfokus pada pendekatan berulang, umpan balik yang konstan dari pemangku kepentingan dan pengembangan program RUP, secara signifikan meningkatkan kualitas program yang dihasilkan. Penelitian tentang pengembangan sistem informasi manajemen kepengasuhan santri mengadopsi metode pengembangan RUP agar sistem yang dihasilkan berkualitas.

Berdasarkan penelitian sebelumnya, sistem informasi manajemen dianggap mampu memudahkan aktor yang terlibat di dalamnya untuk menjalankan proses transaksi operasional jika sistem yang dikembangkan sesuai dengan kebutuhan penggunanya. Proses analisis kebutuhan pengguna merupakan tahap yang penting dalam membangun suatu sistem informasi karena dapat menentukan kesuksesan sistem yang dibangun. Dengan metode pengembangan perangkat lunak RUP dapat meningkatkan kualitas perangkat lunak yang dihasilkan jika proses yang dilakukan berjalan dengan benar.

2.2 Tazkia *International Islamic Boarding School*

Tazkia *International Islamic Boarding School* (IIBS) merupakan sekolah dengan sistem pendidikan Islam yang menerapkan gaya pesantren modern. Tazkia IIBS beralamat di jalan Tirto Sentono nomor 15 Desa Landungsari, Kecamatan Dau, kabupaten Malang, Jawa Timur. Tazkia IIBS merupakan lembaga pendidikan berbasis pesantren modern yang menyelenggarakan sistem pendidikan secara menyeluruh dan berimbang dalam ilmu agama dan ilmu umumnya. Dalam rangka memberikan pendidikan yang terstruktur dan terprogram dengan baik, Tazkia IIBS menetapkan visi dan misinya untuk mencapai tujuan dan cita-citanya.

Tazkia IIBS berupaya maksimal mewujudkan visi dan misi dengan menyelenggarakan beberapa program unggulan, salah satunya program kepengasuhan. Menurut kepala kepesantrenan Tazkia IIBS, program kepengasuhan terdapat dua proses utama yaitu kemandirian dan peribadatan. Kemandirian adalah penilaian terhadap kualitas dan kuantitas kemandirian yang dilakukan santri selama mondok di Tazkia IIBS. Sedangkan peribadatan adalah penilaian terhadap kualitas dan kuantitas aktivitas ibadah santri.

Penilaian kemandirian dan peribadatan dilakukan oleh pengasuh atau jika di Tazkia IIBS dikenal dengan sebutan murabbi/murabbiah. Proses tersebut dimulai ketika awal semester kepala kepesantrenan melakukan pembagian santri ke kamar-kamar yang sudah ditentukan. Setiap kamar berisi sekitar delapan santri. Selain itu kepala kepesantrenan juga membagi murabbi untuk setiap kamar. Biasanya setiap murabbi bertanggung jawab sekitar tiga kamar atau dengan rasio 1:24. Setiap murabbi tersebut akan menjadi pengasuh santri-santri kamar yang dimilikinya selama satu semester.

Setiap hari murabbi bertugas mencatat keterlaksanaan aktivitas santri pada sebuah jurnal yang sudah dibuat oleh kepala kepesantrenan sebelumnya. Jurnal tersebut berisi aktivitas-aktivitas yang dinilai dalam proses kemandirian maupun peribadatan. Beberapa contoh aktivitas kemandirian diantaranya menata baju, memotong kuku, belajar malam dan mandi pagi/sore. Sedangkan untuk peribadatan misalnya semua solat fardhu, solat tahajud dan ibadah lainnya. Penilaian tersebut digunakan sebagai nilai kuantitas santri. Selanjutnya ada nilai ujian yang dilakukan sekali sebagai nilai kualitas. Semua nilai dari jurnal harian dan ujian di rekapitulasi oleh murabbi, kemudian dimasukkan ke ledger penilaian. Ledger penilaian adalah kumpulan nilai hasil rekapitulasi jurnal harian dan nilai ujian. Ledger terebut dilaporkan ke kepala kepesantrenan setiap akhir semester.

2.3 Sistem Informasi Manajemen

Menurut Sutabri (2012), sistem informasi adalah pertemuan kebutuhan pemrosesan transaksi sehari-hari yang mendukung fungsi manajemen operasi organisasi, dengan kegiatan strategis organisasi untuk memberikan laporan yang diperlukan kepada pihak eksternal tertentu. Lebih lanjut, Yakub (2012) menjelaskan bahwa sistem informasi adalah kombinasi terorganisir dari sekelompok orang, perangkat lunak, jaringan komunikasi, perangkat keras, dan

sumber daya yang mengumpulkan, mengubah, dan menyebarkan informasi dalam suatu organisasi.

Menurut O'Brien (2002), Sistem informasi manajemen adalah sistem terintegrasi yang menyediakan informasi yang mendukung kegiatan operasional organisasi, manajemen, dan fungsi pengambilan keputusan. Menurut Kadir (2003), sistem informasi manajemen adalah sistem informasi yang digunakan untuk menyajikan informasi yang digunakan untuk mendukung operasi, manajemen, dan pengambilan keputusan dalam suatu organisasi. Dari uraian tersebut, sistem informasi manajemen adalah suatu sistem yang dirancang untuk menyediakan informasi untuk mendukung pengambilan keputusan mengenai kegiatan administrasi dalam suatu organisasi.

Berdasarkan penjelasan dari para ahli, sistem informasi manajemen pengasuhan santri berarti sistem informasi yang dirancang untuk mendukung operasional kegiatan pengasuhan santri serta menyediakan informasi guna pengambilan keputusan pada kegiatan manajemen. Pada kegiatan kepengasuhan santri di Tazkia IIBS, kegiatan operasional berupa aktivitas mencatat kegiatan ibadah dan kemandirian harian santri. Sedangkan pada level manajerial, pengambilan keputusan berdasarkan informasi yang dihasilkan dari proses operasional kegiatan kepengasuhan.

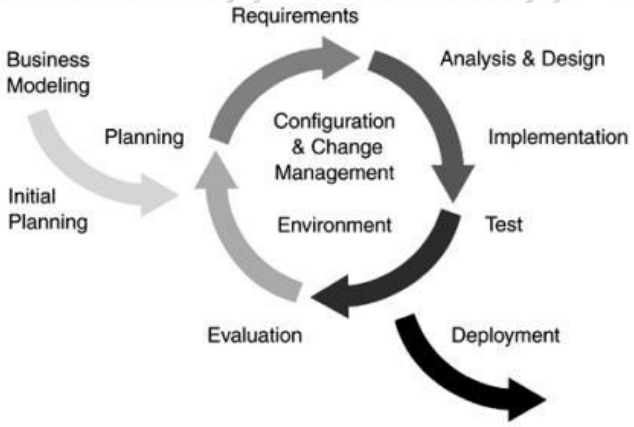
2.4 *Rational Unified Process (RUP)*

Rekayasa perangkat lunak adalah disiplin teknik yang mencakup semua aspek produksi perangkat lunak (Sommerville, 2011). Rekayasa perangkat lunak adalah ilmu yang menjelaskan perangkat lunak dari analisis kebutuhan, desain, pemodelan, implementasi, pengujian dan pemeliharaan perangkat lunak. Rekayasa perangkat lunak terdiri dari proses, seperangkat metode dan seperangkat alat yang memungkinkan para profesional untuk menulis program komputer berkualitas tinggi (Pressman, 2010).

Rational Unified Process (RUP) adalah kerangka proses pengembangan perangkat lunak yang dikembangkan oleh *Rational Software Corporation*, sebuah divisi dari IBM (Anwar, 2014). RUP adalah metodologi rekayasa perangkat lunak yang menyediakan akses untuk menetapkan tugas dan tanggung jawab kepada organisasi pengembang. Hal ini bertujuan untuk memastikan kebutuhan pengguna akhir terpenuhi dan produksi perangkat lunak berkualitas tinggi. RUP menggunakan konsep berorientasi objek yang kegiatannya fokus pada pengembangan model dengan menggunakan *Unified Model Language (UML)*. Metode RUP adalah metode pengembangan yang berorientasi pada proses. Fokus kegiatan pengembangan perangkat lunak di RUP adalah pada pengembangan model dengan UML.

RUP menggunakan pendekatan iteratif. Artinya, urutan fase berulang, seperti yang ditunjukkan pada Gambar 2.1. Setiap iterasi berisi disiplin pengembangan perangkat lunak, dan setiap iterasi juga mendefinisikan tujuannya dan menghasilkan hasil implementasi sistem. Setiap iterasi selanjutnya bergantung

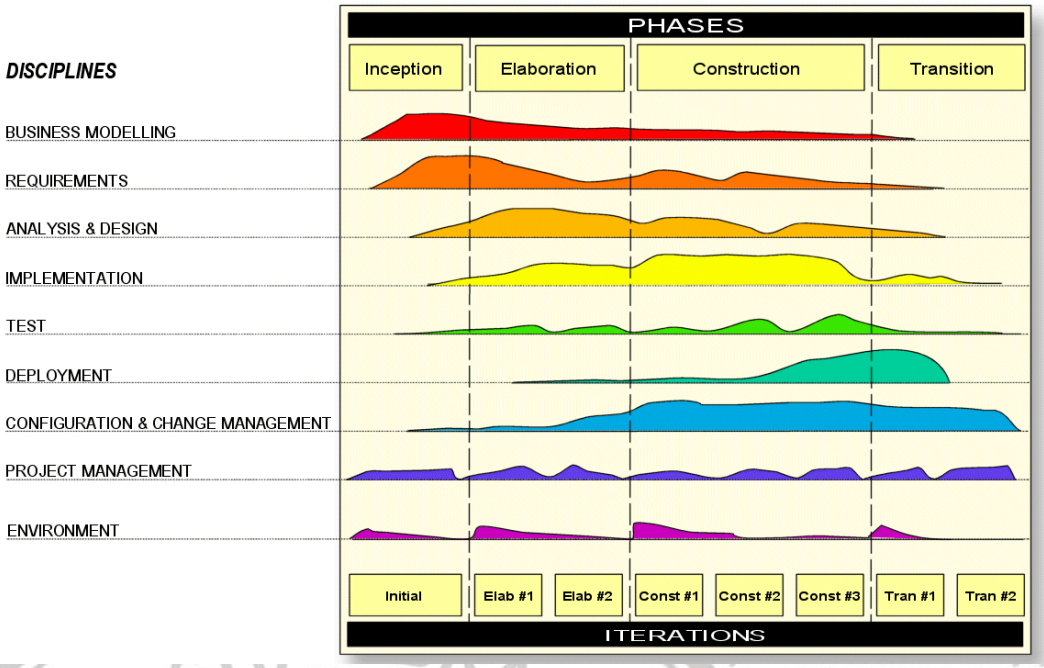
pada hasil iterasi sebelum melanjutkan pembaruan sistem hingga produk akhir selesai dibuat.



Gambar 2.1 Pengembangan iteratif pada *Rational Unified Processs (RUP)*

RUP terdiri dari dua dimensi dapat dilihat pada Gambar 2.2. Dua dimensi dalam RUP yaitu dimensi vertikal dan horizontal. Dimensi vertikal merepresentasikan aspek statis dari proses yang dijelaskan dalam komponen proses: aktivitas, disiplin, artefak, dan peran. Dimensi horizontal mewakili aspek dinamis dari pengembangan perangkat lunak. Hal ini dijelaskan dalam beberapa fase dengan tonggak utama sebagai tanda berakhirnya fase. Setiap fase terdiri dari satu atau lebih iterasi. Setiap tingkat iterasi menanggapi perubahan persyaratan perangkat lunak. Sistem informasi manajemen pengasuhan santri akan dikembangkan secara bertahap dan memungkinkan adanya perubahan kebutuhan dan penambahan fitur. Dimensi horizontal ini terdiri atas *Inception, Elaboration, Construction dan Transition*.

1. *Inception* adalah fase awal pengembang mendefinisikan batas-batas aktivitas, menentukan proses bisnis, menganalisis kebutuhan pengguna dan melakukan perancangan awal.
2. *Elaboration* adalah tahap kedua setelah *inception*. Pada tahap ini, kebutuhan pengguna dianalisis dan dievaluasi, dan dapat terjadi perubahan jika diperlukan atau membutuhkan fitur tambahan. Analisis kebutuhan adalah kegiatan pertama yang dilakukan selama fase penyempurnaan ini. Pada fase ini, kebutuhan fungsional dan non-fungsional ditetapkan. Dokumen definisi kebutuhan ini nantinya akan digunakan sebagai acuan untuk perancangan sistem. Ada juga banyak aktivitas desain pada tahap ini, termasuk diagram *use case*, diagram aktivitas, diagram urutan, diagram kelas, dan representasi lain dari fungsi sistem yang dikembangkan dengan UML.



Gambar 2.2 Arsitektur *Rational Unified Process*

3. *Construction* meliputi desain sistem, pemrograman dan pengujian. Pada fase ini, komponen sistem dikembangkan dan diintegrasikan secara paralel. Pada akhir fase ini, sistem perangkat lunak dan dokumentasi terkait dibuat dan dapat dikirim ke pengguna.
4. *Transition* adalah tahap akhir dari RUP, memigrasikan sistem dari tahap konstruksi ke pengguna dan menjalankannya di lingkungan nyata. Bagian ini diabaikan di sebagian besar model operasi perangkat lunak, tetapi sebenarnya ini adalah aktivitas yang mahal dan terkadang bermasalah. Pada akhir tahap ini, perlu membuat sistem perangkat lunak terdokumentasi yang berfungsi dengan baik di lingkungan operasi.

2.4.1 Pemodelan Proses Bisnis

Menurut Weske (2012), proses bisnis adalah serangkaian kegiatan yang dikoordinasikan dan dilakukan dalam lingkungan organisasi dan teknis. Kegiatan ini digunakan untuk mencapai tujuan bisnis. Setiap proses bisnis ditentukan oleh satu organisasi tetapi dapat berinteraksi dengan operasi organisasi lain.

Proses bisnis adalah serangkaian tindakan yang saling terkait yang bertujuan untuk memecahkan masalah tertentu. Proses bisnis dapat dibagi menjadi beberapa sub-proses, yang masing-masing memiliki karakteristiknya sendiri, tetapi juga membantu untuk mencapai tujuan dari proses utama. Analisis proses bisnis biasanya berkisar pada proses pemetaan dan sub-prosesnya ke tingkat aktivitas.

Business Process Management (BPM) adalah pendekatan yang meningkatkan efisiensi dan efektivitas melalui otomatisasi proses dan kelincahan manajemen perubahan. BPM membantu perusahaan memantau dan mengendalikan semua

elemen proses bisnis mereka, termasuk karyawan, pelanggan, pemasok, dan alur kerja. BPM meningkatkan kualitas proses bisnis melalui mekanisme umpan balik yang lebih baik. Inspeksi *real-time* yang berkelanjutan memungkinkan perusahaan untuk mengidentifikasi masalah dan menyelesaikannya lebih cepat daripada masalah yang lebih besar datang.

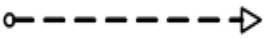
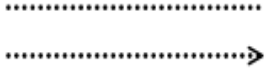
2.4.1.1 Business Process Model and Nation (BPMN)

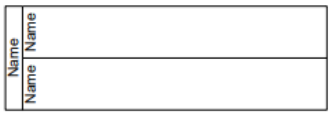
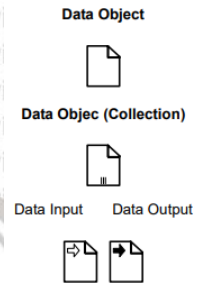
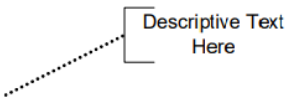
Menurut Weske (2012) *Business Process Modeling Notation* (BPMN) adalah notasi grafis yang menggambarkan logika langkah-langkah dalam proses bisnis. Notasi ini dirancang khusus untuk mengkoordinasikan urutan operasi dan pesan yang mengalir antar aktor dalam aktivitas yang berbeda. BPMN adalah simbol pemodelan proses bisnis yang dikembangkan oleh *Object Management Group*. Tujuan utama BPMN adalah untuk memberikan notasi yang mudah dipahami oleh semua pengguna bisnis, mulai dari analis bisnis yang membuat konsep proses pertama hingga pengembang teknologi yang bertanggung jawab untuk mengimplementasikan proses, manajemen, dan kontrol yang ada. Oleh karena itu, BPMN bertindak sebagai jembatan antara desain proses bisnis dan implementasi proses bisnis.

Variasi dan informasi tambahan dapat ditambahkan dalam kategori elemen dasar untuk mendukung kebutuhan akan kompleksitas tanpa mengubah tampilan dasar diagram. Ada lima kategori dasar item BPMN yaitu aliran objek, data, penghubung objek, *swimlanes* dan *artifacts.* Setiap elemen memiliki bentuk notasi dan tujuan yang berbeda. Elemen dasar pada BPMN dapat dilihat pada Tabel 2.1.

Tabel 2.1 Elemen Dasar BPMN

Elemen	Deskripsi	Notasi
Event	Event adalah sesuatu yang terjadi selama proses berlangsung. Event yang terjadi pada awal proses disebut peristiwa *(g*aris sempit tipis). *Event* yang terjadi di tengah proses (antara kegiatan) *intermediate event* (garis ganda tipis). Event yang terjadi pada akhir proses disebut *End event* (tebal garis tunggal)	Start Intermediate End
Task	Sebuah tugas mewakili satu unit bisnis yang tidak dapat dipecah menjadi tingkat yang lebih rinci dari detail proses bisnis. Tugas adalah	

	tingkat aktivitas terendah yang ditunjukkan dalam diagram proses.	
Manual task	Tugas manual adalah tugas yang dilakukan tanpa bantuan aplikasi.	
Gateway	Sebuah *gateway* diwakili dengan bentuk berlian dan menentukan percabangan dan penggabungan jalur, tergantung pada kondisi yang diinginkan.	
Sequence Flow	Aliran urutan diwakili oleh garis penuh dan panah yang menunjukkan urutan aktivitas yang dilakukan.	
Message Flow	Alur pesan ditunjukkan oleh garis putus-putus, lingkaran terbuka di awal, dan panah terbuka di akhir. Ini memberitahu kita pesan mana yang mengalir melintasi batas-batas organisasi. Alur pesan tidak dapat digunakan untuk menghubungkan aktivitas atau peristiwa dalam grup yang sama.	
Assosiacion	Diwakili oleh garis putus-putus. Ini digunakan untuk mengaitkan artefak atau teks dengan objek aliran dan panah terbuka dapat digunakan untuk menunjukkan arah aliran	
Pool	Mewakili pelaku utama dalam proses dan biasanya memisahkan organisasi yang berbeda. *Pool* memiliki satu atau lebih jalur.	

Lane	Ini digunakan untuk mengatur dan mengategorikan aktivitas dalam grup berdasarkan fungsi atau peran, dan direpresentasikan sebagai persegi panjang yang memperluas lebar atau tinggi grup. Koridor berisi objek aliran, objek koneksi, dan artefak.	
Data object	Objek data menunjukkan kepada pembaca data mana yang diperlukan atau dihasilkan dalam suatu aktivitas	
Group	Grup diwakili dengan persegi panjang bersudut membulat dan garis putus-putus. Grup digunakan untuk mengelompokkan aktivitas yang berbeda tetapi tidak mempengaruhi aliran dalam diagram	
Text Annotation	anotasi digunakan untuk memberikan informasi yang dapat dimengerti kepada pembaca model/diagram	

2.4.2 Analisis Kebutuhan

Persyaratan kebutuhan sistem perangkat lunak menjadi dasar apa yang harus dilakukan sistem dan menetapkan batasan operasional dan implementasi untuk mengomunikasikan dengan benar semua fungsionalitas yang disediakan (Sommerville, 2011). Analisis persyaratan menciptakan spesifikasi untuk sifat operasi perangkat lunak, menampilkan antarmuka antara perangkat lunak dan elemen sistem lainnya, dan mendefinisikan kondisi batas yang harus dipenuhi perangkat lunak (Pressman, 2010). Analisis kebutuhan perangkat lunak adalah proses untuk mendapatkan informasi, model, dan spesifikasi sistem yang dibutuhkan pengguna (Simarmata, 2010). Analisis kebutuhan adalah tahap

pertama dari pengembangan perangkat lunak dan memainkan peran yang sangat penting dalam melanjutkan tahap berikutnya.

Banyak faktor penentu dalam melakukan analisis kebutuhan, seperti kurangnya pemahaman pengguna tentang pengetahuan sebenarnya tentang komputer, teknologi pemrograman, dan entitas TI. Memiliki persyaratan yang jelas dan benar yang diinginkan pengguna adalah langkah pertama yang baik untuk membantu membangun perangkat lunak di langkah berikutnya. Analisis persyaratan menciptakan spesifikasi untuk sifat operasi perangkat lunak, menampilkan antarmuka antara perangkat lunak dan elemen sistem lainnya, dan mendefinisikan kondisi batas yang harus dipenuhi perangkat lunak. Kebutuhan sistem dapat dibagi menjadi kebutuhan fungsional dan non-fungsional (Sommerville, 2011).

Persyaratan fungsional adalah pernyataan tentang layanan yang harus diberikan ke sistem untuk merespons input tertentu dan melakukan operasi dalam situasi tertentu. Persyaratan ini perlu dijelaskan secara rinci pada setiap tingkat sistem. Untuk kebutuhan non-fungsional, penekanannya adalah pada pembatasan layanan atau fitur yang disediakan oleh sistem. Dokumen manifes non-fungsional ini mencakup batasan waktu, proses pengembangan, dan standarisasi keluaran sistem. Secara umum, persyaratan ini berasal dari persyaratan pengguna sistem, seperti produk, organisasi, dan persyaratan eksternal.

Analisis kebutuhan perangkat lunak dapat dipecah menjadi lima area kerja: identifikasi masalah, penilaian dan integrasi, pemodelan, spesifikasi dan kontrol spesifikasi (Pressman, 2010). Analisis kebutuhan perangkat lunak selalu dimulai dengan komunikasi antara dua atau lebih bagian melalui percakapan. Gause dan Weinberg (1989) menyarankan bahwa analis pertama-tama mengajukan pertanyaan tata bahasa yang sederhana dalam konteks dan mengarah pada pemahaman yang lebih mendasar (Pressman, 2010).

2.4.2.1 Wawancara

Metode survei adalah jenis studi di mana orang terbiasa menerima informasi. Oleh karena itu, perlu dikembangkan alat penelitian: kuesioner (daftar pertanyaan) dan pedoman wawancara (Hasibuan, 2007). Wawancara tatap muka memiliki beberapa keuntungan. Artinya pewawancara dapat memperkuat kerjasama antara pewawancara dan responden dan memperjelas pertanyaan secepat mungkin. Secara fisik, wawancara dapat dibagi menjadi dua bagian.

1. Wawancara terstruktur: terdiri dari daftar pertanyaan yang harus dijawab oleh pewawancara saja. Dalam hal ini terlihat seperti kuisioner, dengan perbedaan responden harus melakukannya secara langsung dengan pewawancara, jika ada yang tidak mengerti maka bertanyalah padanya dan pewawancara dapat langsung mengecek kelengkapan jawaban responden.
2. Wawancara tidak terstruktur: wawancara yang dilakukan oleh pewawancara secara sukarela, tetapi pewawancara tetap mengacu pada data atau informasi yang diperlukan. Dalam hal ini, pewawancara juga dapat menggunakan instruksi yang hanya menguraikan apa yang harus ditanyakan.

Wawancara bukanlah tugas yang mudah. Dalam hal ini, pewawancara harus dapat membangun suasana santai namun tetap serius agar narasumber siap menjawab pertanyaan dengan jujur.

2.4.3 Perancangan Sistem

Menurut Sommerville (2011), perancangan sistem adalah proses mengembangkan model abstrak sebuah sistem, yang mana setiap model menggambarkan sudut pandang atau perspektif yang berbeda dari sebuah sistem.

Menurut Pressman (2010), perancangan perangkat lunak adalah proses berulang di mana persyaratan diterjemahkan ke dalam "cetak biru" untuk membangun perangkat lunak. Dengan kata lain, perancangan dinyatakan sebagai abstraksi tingkat tinggi, tingkat yang dapat ditelusuri langsung ke tujuan sistem tertentu, data yang lebih rinci, persyaratan fungsional dan perilaku. Saat iterasi perancangan terjadi, penyempurnaan selanjutnya menghasilkan representasi rancangan pada tingkat abstraksi yang lebih rendah.

Sebelum langkah selanjutnya, pengembang dapat menilai kualitas hasil desain dengan menganalisis kesalahan, inkonsistensi dan kelalaian, menemukan alternatif yang lebih baik, dan menganalisis apakah model desain sesuai dengan jadwal, biaya, dan batasan. McGlaughlin dalam Pressman (2010) mengemukakan tiga karakteristik yang menjadi pedoman untuk mengevaluasi desain yang baik, yaitu:

1. Rancangan harus mengimplementasikan semua persyaratan eksplisit dari model persyaratan dan memperhitungkan semua persyaratan implisit yang diinginkan oleh para pemangku kepentingan.
2. Rancangan harus menjadi panduan yang mudah dibaca, mudah dipahami bagi mereka yang membuat kode dan kemudian menguji dan mendukung perangkat lunak.
3. Rancangan harus memberikan gambaran lengkap dari perangkat lunak dari perspektif implementasi, meliputi domain data, fungsional, dan operasional.

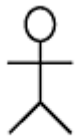
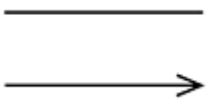
Pemodelan sistem umumnya menggambarkan sistem menggunakan semacam notasi grafis (Sommerville, 2011). Saat ini, hampir seluruh pemodelan berdasarkan pada notasi *Unified Modelling Language (UML)*. UML adalah bahasa standar yang digunakan untuk mendefinisikan, memvisualisasikan, mengatur, dan mendokumentasikan elemen-elemen perangkat lunak yang akan dikembangkan (Pressman, 2010). UML memiliki banyak jenis diagram dan juga mendukung pembuatan berbagai macam tipe pemodelan sistem. Namun, analisis dan desain sistem berorientasi objek (OOSA & D) menggunakan diagram *use case* (bersama dengan deskripsi *use case* yang sesuai), diagram urutan (*sequence diagram*), dan diagram kelas (*class diagram*) sebagai model utama untuk menggambarkan model analisis sistem perangkat lunak (George et al, 2004).

2.4.3.1 *Use case diagram*

Use case dapat didefinisikan sebagai skenario sederhana yang mendeskripsikan apa yang pengguna harapkan dari sebuah sistem (Sommerville, 2011). *Use case* didefinisikan dari sudut pandang aktor. Aktor adalah peran yang dimainkan orang (pengguna) atau perangkat saat mereka berinteraksi dengan perangkat lunak. Diagram *use case* menggambarkan apa yang dilakukan bukan bagaimana melakukannya. Diagram *Use case* dekat kaitannya dengan kejadian-kejadian. Kejadian (*scenario*) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem.

Menurut Pressman (2010), langkah pertama dalam membuat *use case* adalah mendefinisikan set "aktor" yang akan terlibat dalam cerita. Aktor adalah orang (atau perangkat) yang berbeda yang menggunakan sistem atau produk dalam konteks fungsi dan perilaku yang akan dijelaskan. Aktor mewakili peran yang dimainkan orang (atau perangkat) saat sistem beroperasi. Didefinisikan secara lebih formal, aktor adalah segala sesuatu yang berkomunikasi dengan sistem atau produk dan yang berada di luar sistem itu sendiri. Setiap aktor memiliki satu atau lebih tujuan saat menggunakan sistem. Dalam banyak kasus, tidak perlu membuat representasi grafis dari skenario penggunaan. Namun, representasi diagram dapat memfasilitasi pemahaman, terutama ketika skenarionya kompleks. UML menyediakan kemampuan diagram use case seperti pada Tabel 2.2.

Tabel 2.2 Notasi *Use Case Diagram*

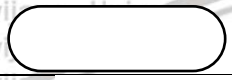
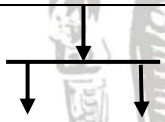
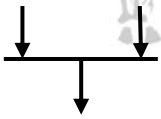
No	Simbol	Deskripsi
1	 An Actor	***Actors.*** Digambarkan dengan bentuk *stick man* serta peran aktor
2	 A Use Case	***Use case***. Setiap use case diwakili oleh oval serta kata kerja yang menjelaskan aktivitas yang dilakukan
3		***Communicates association***. Penghubung antara aktor dan *use case* yang berinteraksi. Simbol panah menunjukkan elemen yang memulai interaksi terlebih dahulu

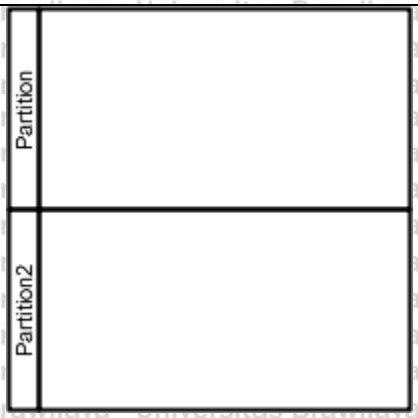
2.4.3.2 Diagram aktivitas (*activity diagram*)

Use case dapat didukung dengan *activity diagram* dengan menyediakan representasi grafis berupa aliran interaksi berdasarkan skenario spesifik (Pressman 2010). Diagram aktivitas dimaksudkan untuk menunjukkan aktivitas yang membentuk suatu proses sistem dan aliran kendali dari satu aktivitas ke

aktivitas lainnya. Awal suatu proses ditunjukkan dengan lingkaran penuh; diakhiri dengan lingkaran penuh di dalam lingkaran lain. Persegi panjang dengan sudut bulat mewakili aktivitas, yaitu sub-proses spesifik yang harus dilakukan. Secara umum, notasi pada *activity diagram* dapat dilihat pada Tabel 2.3**Error! Reference source not found.**.

Tabel 2.3 Notasi *Activity Diagram*

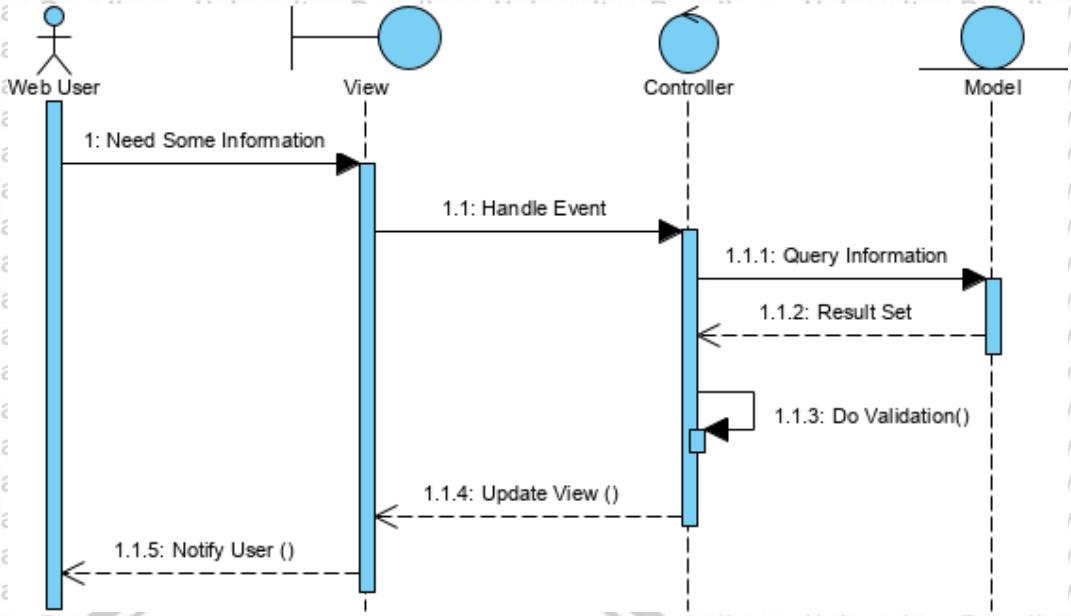
No	Simbol	Deskripsi
1		***Initial state***. Menggambarkan awal dari serangkaian tindakan atau kegiatan
2		***Actions***. Sebuah tugas yang harus dilakukan
3		***Flow***. Menunjukkan urutan eksekusi
4		***Decisions***. Mewakili kondisi pengujian untuk memastikan bahwa aliran kontrol atau aliran objek hanya turun satu jalur
5		***Merge***. Menyatukan kembali jalur keputusan berbeda yang dibuat menggunakan simpul keputusan.
6		***Fork***. Membagi perilaku menjadi serangkaian aktivitas (atau tindakan) paralel atau bersamaan
7		***Join***. Menyatukan kembali serangkaian aktivitas (atau tindakan) paralel atau bersamaan
8		***Final state***. Menghentikan semua aliran kontrol dan aliran objek dalam suatu aktivitas

9		***Swimline and partition*** adalah cara untuk mengelompokkan aktivitas yang dilakukan oleh aktor yang sama pada diagram aktivitas atau mengelompokkan aktivitas dalam satu rangkaian

2.4.3.3 *Sequence diagram*

Menurut Sommerville (2011) diagram aliran *(sequence diagram)* pada UML digunakan untuk memodelkan interaksi antara aktor dan objek dalam suatu sistem dan antara objek itu sendiri. Sesuai dengan namanya, *sequence diagram* menunjukkan urutan interaksi yang terjadi selama *use case* tertentu. MVC *(Model-view-controller)* adalah kerangka kerja perangkat lunak yang populer untuk menghubungkan antarmuka pengguna dengan model data (Visual-paradigm.com). Kerangka kerja MVC umumnya menganggap aplikasi memiliki tiga lapisan utama: presentasi (UI), logika aplikasi, dan manajemen sumber daya. Pola *model-view-controller* menyediakan tiga komponen atau objek utama untuk digunakan dalam pengembangan perangkat lunak:

1. *Model*: merepresentasikan struktur data dan logika yang mendasari perangkat lunak
2. *View*: komponen View digunakan untuk semua logika antar muka perangkat lunak yang berinteraksi dengan pengguna akhir
3. *Controller*: mewakili kelas yang menghubungkan model dan *view* serta digunakan untuk berkomunikasi antara kelas dalam model dan *view*.

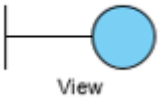
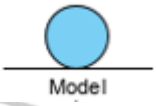
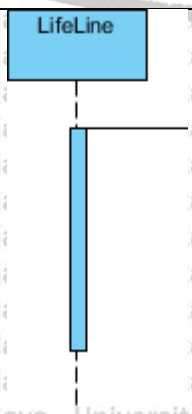


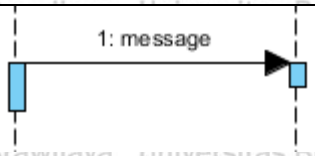
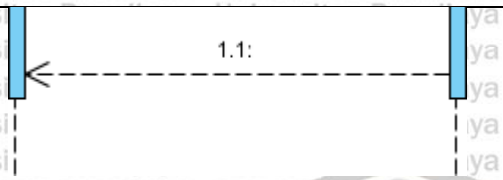
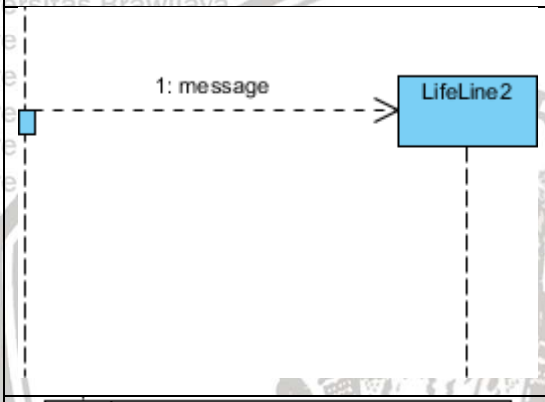
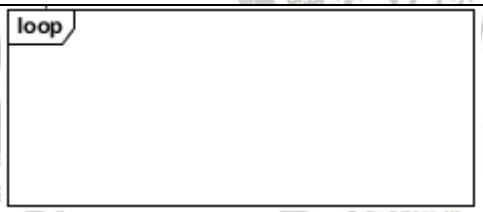
Gambar 2.3 Contoh *sequence diagram*

Sumber: (visual-paradigm.com)

Gambar 2.3 adalah contoh *sequence diagram* dengan menggunakan konsep MVC. UML memiliki berbagai macam notasi untuk diagram urutan yang memungkinkan untuk memodelkan berbagai macam jenis reaksi. Tabel 2.4 adalah notasi yang terdapat pada *sequence diagram*.

Tabel 2.4 Notasi *Sequence Diagram*

Simbol	Deskripsi
	**Aktor** Komponen yang berbentuk stick figure. Mewakili peran yang dijalankan oleh pengguna manusia, perangkat eksternal, atau objek lainnya.
	***Lifeline*** Merepresentasikan entitas individu dalam Interaksi.
	***View*** Merepresentasikan objek antarmuka pengguna
	***Controller*** Merepresentasikan objek yang menghubungkan antara *view* dan *model*
	***Model*** Merepresentasikan entitas yang mendasari logika dan struktur data
	***Activations*** Persegi panjang pada garis hidup mewakili periode selama elemen melakukan operasi.

	***Call Message*** Sebuah pesan mendefinisikan komunikasi tertentu antara garis hidup suatu Interaksi. *Call message* adalah semacam pesan yang diminta untuk dioperasikan oleh target yang dituju
	***Return Message*** Pesan balasan adalah pesan yang mewakili penyampaian informasi kembali ke pemanggil dari pesan sebelumnya yang sesuai.
	***Create Message*** *Create message* adalah pesan yang merepresentasikan instansiasi dari *lifeline*
	***Fragments*** Fragmen adalah pengelompokan logika yang direpresentasikan sebagai persegi panjang, yang berisi struktur kondisional yang mempengaruhi aliran pesan.

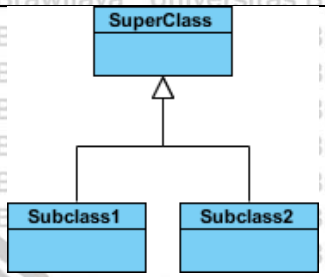
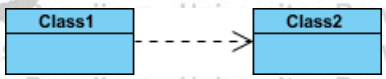
Sumber: Diadaptasi dari visual-paradigm.com

2.4.3.4 *Class diagram*

Class diagram menggambarkan struktur kelas dalam sistem dan hubungan di antara kelas-kelas tersebut (Sommerville, 2011). Dalam praktiknya, untuk mengidentifikasi objek dalam sistem harus menggunakan beberapa sumber pengetahuan untuk menemukan kelas objek. Kelas objek, atribut, dan operasi yang awalnya diidentifikasi dari deskripsi sistem informal dapat menjadi titik awal untuk desain. Informasi lebih lanjut dari pengetahuan domain aplikasi atau analisis skenario kemudian dapat digunakan untuk memperbaiki dan memperluas objek awal. Informasi ini dapat dikumpulkan dari dokumen persyaratan, diskusi dengan pengguna, atau dari analisis sistem yang ada.

Secara garis besar, *class* dapat diartikan sebagai gambaran umum dari satu jenis objek dalam sistem. Dalam rekayasa perangkat lunak, *class diagram* dalam UML adalah jenis diagram struktur statis yang menggambarkan struktur sistem dengan menunjukkan kelas sistem, atributnya, operasi (atau metode), dan hubungan antar objek. Beberapa hubungan yang ada pada *class diagram* ditunjukkan pada Tabel 2.5.

Tabel 2.5 Hubungan pada *Class Diagram*

Tipe hubungan	Definisi	Notasi
Inheritance atau generalisasi	Mewakili hubungan yang bermakna "adalah sebuah". Nama kelas abstrak ditampilkan dalam huruf miring. SubClass1 dan SubClass2 adalah spesialisasi dari Super Class. Digambarkan dengan garis padat dengan panah berongga yang menunjuk dari anak ke kelas induk	
Asosiasi	Sebuah penghubung antara dua kelas. Dinotasikan dengan garis solid yang menghubungkan dua kelas	
Dependency	Hubungan antara dua kelas yang mana ketika ada perubahan pada satu *class* dapat menyebabkan perubahan pada yang lain (tetapi tidak sebaliknya). Class1 tergantung pada Class2. Digambarkan dengan garis putus-putus dengan panah terbuka	

Sumber: Diadaptasi dari visual-paradigm.com

2.4.4 Implementasi Sistem

Menurut Sommerville (2011), implementasi adalah proses mewujudkan desain menjadi program. Implementasi sistem berkaitan dengan mengubah spesifikasi kebutuhan menjadi sistem perangkat lunak yang dapat dieksekusi. Pembuatan kode otomatis dari model desain membantu mempercepat proses ini. Menurut Sommerville (2011), menggunakan UML untuk mendokumentasikan desain tepat dilakukan jika pemrograman berorientasi objek.

2.4.4.1 *Object Oriented Programming* (OOP)

Pemrograman berorientasi objek adalah paradigma pengembangan sistem yang memandang segala sesuatu sebagai objek dan memiliki atribut dan sifat yang unik (Shalahudin & S, 2015). Keuntungan mengembangkan sistem dengan pemrograman berorientasi objek adalah:

1. Dapat digunakan kembali (*reusable*), sehingga meningkatkan produktivitas.
2. Kemudahan perawatan karena penggunaan objek.
3. Adanya konsistensi menggunakan konsep pewarisan (*inheritance*).
4. Pengembangan didasarkan pada kejadian nyata yang meningkatkan kualitas perangkat lunak dan memenuhi kebutuhan pengguna.

2.4.4.2 Codeigniter

Codeigniter adalah sebuah kerangka kerja dengan model MVC (Model, View, Controller) untuk membangun website dinamis dengan menggunakan PHP (Supono dan Putratama, 2016). Codeigniter memudahkan pengembang web untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuat dari awal. Kelebihan CodeIgniter (CI) dibandingkan dengan kerangka kerja lain:

1. Konfigurasi yang sedikit (*nearly zero configuration*): Untuk menggunakan codeigniter dengan pengaturan yang standar, hanya perlu mengubah sedikit *file* pada folder config.
2. Menggunakan konsep MVC: Pengerjaan antara logika dengan *layout* telah dipisahkan, sehingga *programmer* dan *designer* dapat mengerjakan tugas masing-masing dengan mudah.
3. Banyak komunitas: Banyaknya komunitas CI memudahkan developer berdiskusi satu sama lain.
4. Dokumentasi lengkap: Setiap paket instalasi Codeigniter sudah disertai *user guide* yang lengkap dan mudah dipahami.

2.4.5 Pengujian Sistem

Pengujian dirancang untuk memeriksa apakah program melakukan apa yang seharusnya dilakukan dan untuk menemukan kesalahan sebelum menggunakan program Sommervile (2011). Proses pengujian berfokus pada pengurangan kesalahan yang terjadi selama pengoperasian sistem informasi dan pada pengujian kualitas sistem informasi. Pengujian sistem terdiri dari pengujian

fungsionalitas dan kualitas sistem informasi. Tes fungsional digunakan untuk memverifikasi kinerja tugas yang berhasil diselesaikan.

Pengujian adalah bagian dari proses verifikasi dan validasi perangkat lunak (V&V) yang lebih luas. Verifikasi dan validasi bukan hal yang sama, walaupun sering kali membingungkan. Barry Boehm, dalam Sommerville (1979) menjelaskan tentang perbedaan verifikasi dan validasi. Validasi adalah "Apakah kita membangun produk yang tepat?". Sedangkan verifikasi "Apakah kita membangun produk dengan benar?".

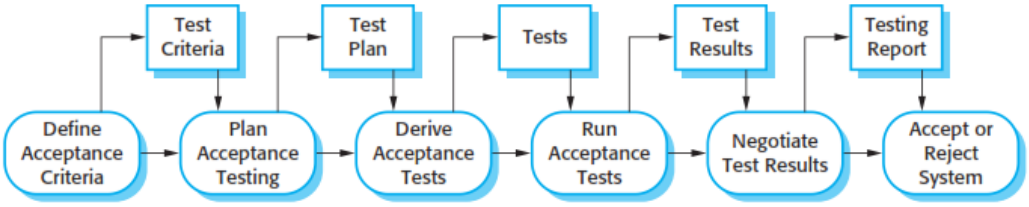
Pengujian kualitas sistem digunakan untuk mengecek apakah sistem yang dibuat sudah layak untuk digunakan (Pressman 2002). Pengujian dapat dilakukan dengan menggunakan beberapa metode antara lain, *White-box testing* dan *Black-box testing*. *White-box testing* digunakan untuk melakukan pengecekan tentang logika internal dan struktur kode, apakah sudah benar (Khan 2012). Teknik ini didasarkan pada informasi tentang bagaimana perangkat lunak telah dirancang atau dikodekan. Sedangkan *black-box testing* digunakan untuk menguji sistem informasi apakah masukan yang diterima dan keluaran sudah beroperasi dengan benar. Pengujian dikategorikan sebagai *black-box* testing jika kasus uji hanya bergantung pada masukan atau keluaran perangkat lunak. Para ahli telah mengembangkan berbagai macam metode atau alat untuk melakukan pengujian.

2.4.5.1 *Validation testing*

Pengujian validasi adalah langkah-langkah pengujian yang dilakukan untuk memastikan bahwa sistem yang dikembangkan dapat beroperasi sesuai dengan persyaratan tertentu (Pressman, 2010). Pengujian berfokus pada tindakan yang terlihat oleh pengguna dan keluaran yang dapat dikenali pengguna dari sistem. Uji validasi dilakukan dengan menguji sekumpulan kasus uji yang dibuat menurut prosedur pengujian yang diberikan. Setelah menjalankan setiap kasus uji sebagai bagian dari uji validasi, terdapat dua kemungkinan hasil. Pertama adalah bahwa fungsi dijalankan sesuai kebutuhan dan dapat diterima, dan yang kedua adalah kesalahan dan cacat ditampilkan di bagian akhir.

2.4.5.2 *User Acceptance Testing*

Menurut Sommerville (2011), pengujian pengguna merupakan tahapan dalam proses pengujian dimana pengguna memberikan masukan dan saran atas pengujian sistem. Pengujian pengguna sangat penting, bahkan ketika pengujian sistem dan rilis yang komprehensif telah dilakukan. Alasannya adalah pengaruh dari lingkungan kerja pengguna memiliki pengaruh besar pada keandalan, kinerja, kegunaan, dan ketahanan sistem. Secara praktik tidak mungkin bagi pengembang sistem untuk mereplikasi lingkungan kerja sistem. Maka pengujian pengguna perlu dilakukan pada lingkungan nyata dimana sistem tersebut akan dijalankan.



Gambar 2.4 Proses *User Acceptance Testing*

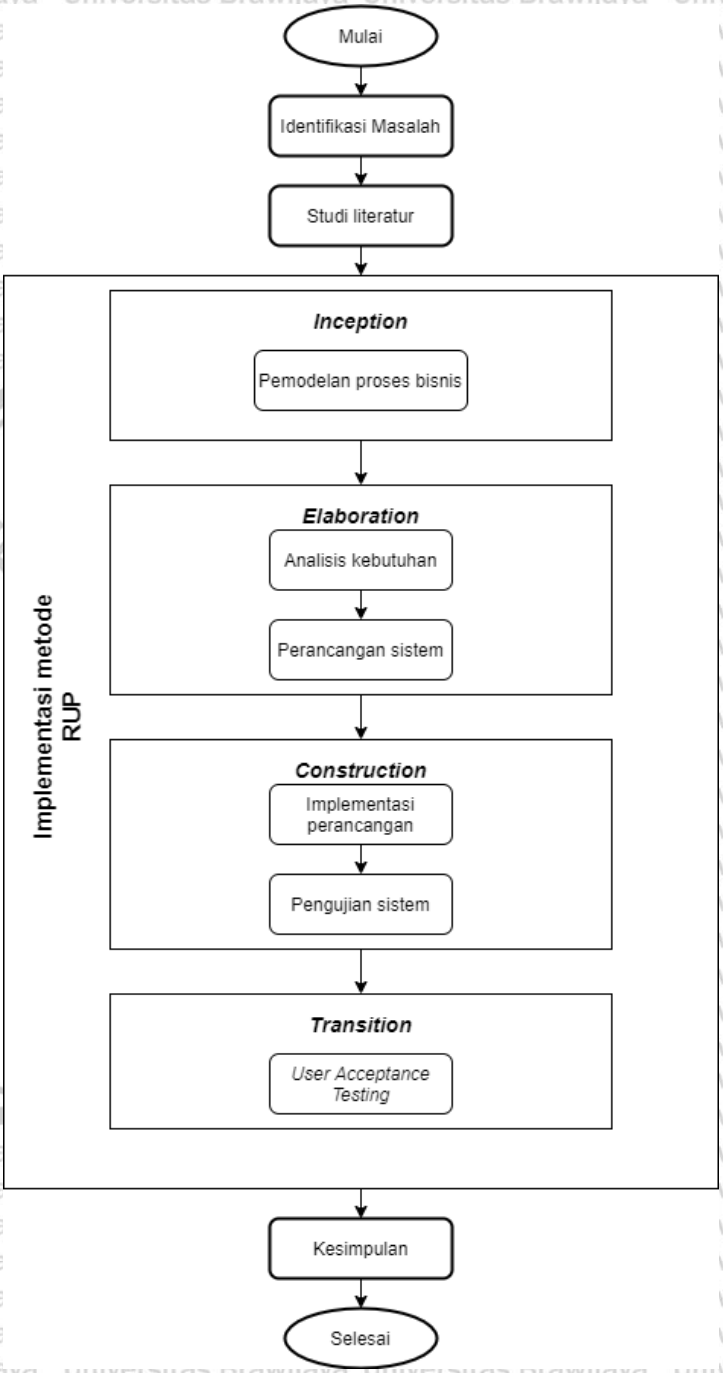
Sumber: (Sommerville, 2011)

Acceptance testing adalah di mana pengguna menguji sistem untuk memutuskan apakah sistem tersebut siap untuk diterima dari pengembang sistem dan digunakan di lingkungan pengguna (Sommerville, 2011). Pengujian ini bertujuan menentukan kesepakatan dengan pengguna dalam menerima atau menolak perangkat lunak yang diberikan. Pengujian ini melibatkan pengguna secara formal menguji sistem untuk memutuskan apakah itu harus diterima atau tidak. Terdapat enam tahapan dalam proses *acceptance testing* seperti yang ditunjukkan pada Gambar 2.4.

1. Mendefinisikan kriteria penerimaan
2. Merencanakan pengujian
3. Membuat kasus uji penerimaan
4. Menjalankan pengujian
5. Negosiasi hasil pengujian
6. Menerima atau menolak sistem

BAB 3 METODOLOGI PENELITIAN

Metodologi penelitian yang akan dilakukan ini melalui beberapa tahapan-tahapan yang digambarkan pada Gambar 3.1 diagram alir penelitian berikut.



Gambar 3.1 Diagram Alir Penelitian

3.1 Identifikasi Masalah

Tahap pertama yang dilakukan adalah mengidentifikasi masalah yang terjadi pada objek penelitian yaitu Tazkia IIBS. Tahapan identifikasi masalah bertujuan untuk mencari permasalahan yang dapat diangkat menjadi penelitian sehingga memunculkan alternatif penyelesaian untuk permasalahan tersebut. Proses identifikasi masalah dilakukan dengan melakukan observasi langsung dan berdiskusi dengan kepala kepesantrenan. Rancangan pertanyaan diskusi dapat dilihat pada LAMPIRAN A. Masalah tersebut terkait dengan proses pengolahan nilai peribadatan dan kemandirian santri serta pelaporan nilai santri dalam kegiatan kepengasuhan.

3.2 Studi Literatur

Studi literatur adalah tahapan mencari sumber literatur yang mendukung penelitian. Sumber literatur dapat berasal dari buku, jurnal, atau penelitian yang telah dilakukan sebelumnya yang berhubungan dengan pengembangan sistem informasi kepengasuhan. Literatur yang digunakan berkaitan dengan profil Tazkia IIBS, sistem informasi, pemodelan proses bisnis, metode pengembangan perangkat lunak RUP, analisis kebutuhan, perancangan, implementasi dan pengujian perangkat lunak. Studi literatur dilakukan untuk mendasari langkah-langkah yang dilakukan dalam penelitian agar sesuai dengan teori atau penelitian-penelitian yang sudah ada.

3.3 Pemodelan Proses Bisnis

Pemodelan proses bisnis dilakukan untuk menggambarkan alur aktivitas bisnis dari proses kepengasuhan di Tazkia. Tujuan dari pemodelan ini untuk mengerti dan memahami setiap langkah dari proses kepengasuhan baik sebelum dan sesudah adanya sistem informasi manajemen kepengasuhan. Pertama yang dimodelkan adalah bisnis proses yang saat ini sedang terjadi (*as-is*). Setelah bisnis proses yang saat ini dimodelkan, kemudian memodelkan bisnis proses yang akan dilakukan (*to-be*). Pemodelan dibantu dengan menggunakan *software Bizagi Modeler*. Proses pemodelan proses bisnis dilakukan dengan menganalisis proses bisnis yang terjadi saat ini dari hasil wawancara yang dilakukan dengan kepala kepesantrenan. Hasil wawancara dapat dilihat pada LAMPIRAN B. Setelah memahami masalah yang terjadi, kemudian dibuat proses bisnis baru untuk memperbaiki permasalahan proses bisnis saat ini.

3.4 Analisis Kebutuhan

Tahap analisis kebutuhan bertujuan untuk melakukan identifikasi terhadap kebutuhan sistem yang akan dibangun untuk mencari apa yang dibutuhkan dan apa yang harus bisa dilakukan sistem berupa kebutuhan fungsional. Metode pengumpulan datanya berasal dari hasil wawancara dan melakukan pengamatan secara langsung. Tipe wawancara yang digunakan adalah wawancara secara langsung dengan kepala Kepesantrenan Tazkia IIBS. Daftar pertanyaan wawancara dapat dilihat pada LAMPIRAN A. Sedangkan, untuk kebutuhan data sekunder

diperoleh dari sejumlah dokumen dan laporan mengenai aktivitas kepengasuhan. Data yang sudah diperoleh tersebut digunakan sebagai dasar persyaratan kebutuhan sistem. Setelah dibuat kebutuhan fungsional kemudian memodelkannya dengan diagram *use case* dan *use case scenario*.

3.5 Perancangan Sistem

Perancangan digunakan sebagai acuan untuk mengimplementasikan sistem, dimana kebutuhan sistem akan didefinisikan lebih rinci pada tahap ini. Tahap perancangan sistem bertujuan untuk menerjemahkan dari analisis kebutuhan yang sudah didapatkan ke dalam perancangan - perancangan sistem, sehingga akan mudah dipahami. Standar yang digunakan dalam perancangan menggunakan standar yang sudah umum digunakan yaitu UML agar mudah dipahami. Dalam proses ini akan dihasilkan model-model perancangan berupa *activity diagram, sequence diagram* dan *class diagram.* Selain itu pada proses ini juga dilakukan pemetaan *class diagram* ke model relasional. Terakhir dilakukan perancangan antar muka sistem.

3.6 Implementasi Sistem

Tahap implementasi sistem adalah untuk membangun sistem sesuai perancangan yang telah dibuat sebelumnya. Hasil dari tahap perancangan diimplementasikan dalam bentuk *website.* Tahap implementasi terdapat tiga bagian yaitu implantasi basis data, kode program dan antar muka. Implementasi basis data merupakan tahap merealisasikan hasil perancangan pada tahap pemetaan *class* ke model relasional. Implementasi basis data adalah membuat objek basis data seperti tabel, indeks, dan batasan (*constrain*) sesuai diagram relasional dengan . Implementasi kode program adalah menulis baris kode dengan bahasa pemrograman *php* sesuai perancangan *class* dan *sequence diagram.* Implementasi kode program dilakukan dengan memanfaatkan *framework codeigniter*. Selain itu dilakukan implementasi antarmuka agar sistem dapat bekerja secara baik. Implementasi antarmuka disesuaikan dengan desain perancangan antarmuka. Keluaran yang dihasilkan dari tahap implementasi ini adalah sistem informasi manajemen pengasuhan santri yang sesuai dengan perancangan.

3.7 Pengujian Sistem

Tahap pengujian sistem termasuk dalam tahap *construction* dalam RUP setelah sistem dibuat. Pengujian sistem bertujuan untuk mengetahui apakah sistem sudah dibangun dengan benar. Tahap pengujian dilakukan dengan metode pengujian *validation testing*. Pengujian *validation testing* dilakukan untuk mengetahui apakah kebutuhan fungsional dari sistem yang telah dibuat berjalan sesuai perancangannya atau tidak. Pengujian *validation testing* ini dilakukan dengan cara mendefinisikan kasus uji atau *test case berdasarkan use case*. Selanjutnya akan dilakukan perbandingan antara hasil pengujian dengan ekspektasi pengujian.

3.8 *User Acceptance Testing*

Setelah melakukan pengujian sistem selanjutnya memindahkan perangkat lunak dari lingkungan produksi ke lingkungan nyata dimana sistem tersebut digunakan. Pada lingkungan nyata dilakukan *user acceptance testing.* Tujuan dari pengujian ini adalah untuk mengetahui perangkat lunak sudah siap dan dapat digunakan oleh pengguna akhir untuk menjalankan fungsi dan tugas yang harus dilakukan oleh perangkat lunak tersebut. Proses pengujian dimulai dari menentukan kriteria penerimaan. Kemudian merencanakan pengujian akan dilakukan seperti apa. Setelah perencanaan pengujian dilakukan, dibuatlah kasus uji dan dapat diujikan ke pengguna. Pengujian dilakukan langsung ke kepala kepesantrenan dan murabbi selaku aktor yang terlibat langsung dengan sistem.

3.9 Kesimpulan

Tahap terakhir dari penelitian ini adalah penutup yang berisi kesimpulan dan saran. Kesimpulan penelitian ini diharapkan mampu menjawab rumusan masalah yang diangkat. Kemudian dari keseluruhan rangkaian penelitian ini dapat digunakan sebagai saran bagi penelitian di masa mendatang.

BAB 4 HASIL DAN PEMBAHASAN

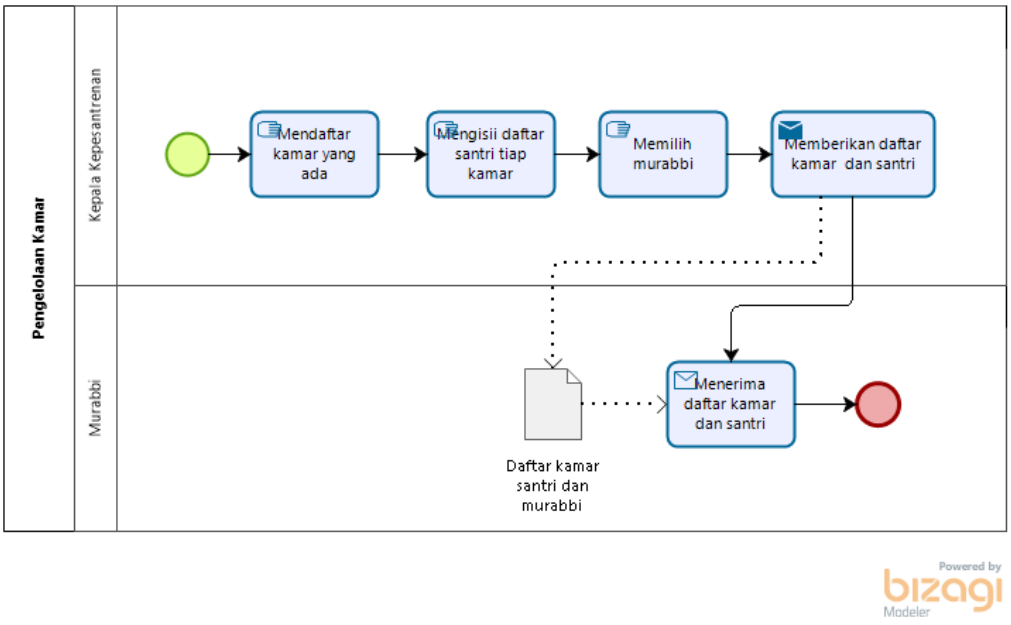
4.1 Pemodelan Proses Bisnis

Pemodelan proses bisnis ini dilakukan untuk mengetahui proses bisnis yang berjalan saat ini, atau disebut sebagai proses bisnis *as-is*. Setelah memodelkan proses bisnis saat ini, kemudian memodelkan proses bisnis usulan atau disebut *to-be* sebagai usulan perbaikan proses bisnis yang saat ini berjalan. Proses bisnis dimodelkan dengan *Business Process Modeling Notation* (BPMN).

4.1.1 Proses Bisnis Saat Ini (*As-is*)

Berdasarkan hasil wawancara dengan kepala kepesantrenan Tazkia IIBS, terdapat tiga proses bisnis yang terjadi pada unit kepengasuhan. Tiga proses tersebut adalah pengelolaan kamar santri dan murabbi, penilaian kemandirian dan penilaian peribadatan. Proses tersebut melibatkan kepala kepesantrenan dan murabbi. Adapun proses bisnis *as-is* dapat digambarkan sebagai berikut:

4.1.1.1 Pengelolaan kamar

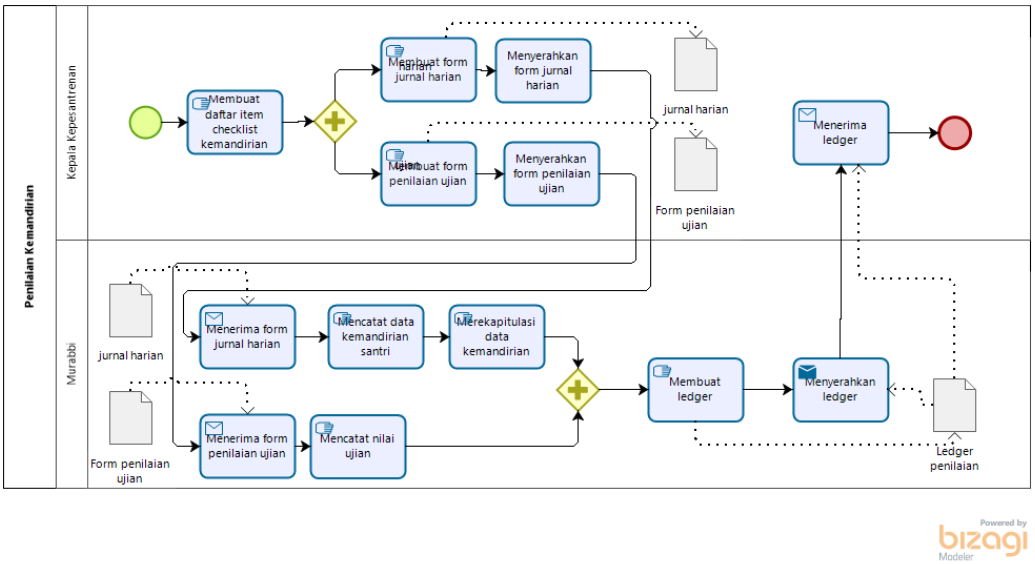


Gambar 4.1 Proses Bisnis *As-is* Pengelolaan Kamar

Gambar 4.1 menunjukkan proses bisnis *as-is* dari kegiatan pengelolaan kamar. Pengelolaan kamar adalah proses mengelola kamar yang akan ditempati oleh santri dan siapa murabbi yang bertanggung jawab atas kamar tersebut. Proses ini dilakukan setiap semester. Proses dimulai dari kepala kepesantrenan mendaftar kamar mana saja yang akan digunakan untuk kamar santri. Selanjutnya membagi santri ke kamar yang ditentukan. Setelah itu kepala kepesantrenan menentukan siapa murabbi dari masing-masing kamar tersebut. Setelah proses tersebut

selesai, kepala kepesantrenan menyerahkan daftar kamar santri dan murabbi kepada semua murabbi.

4.1.1.2 Penilaian Kemandirian

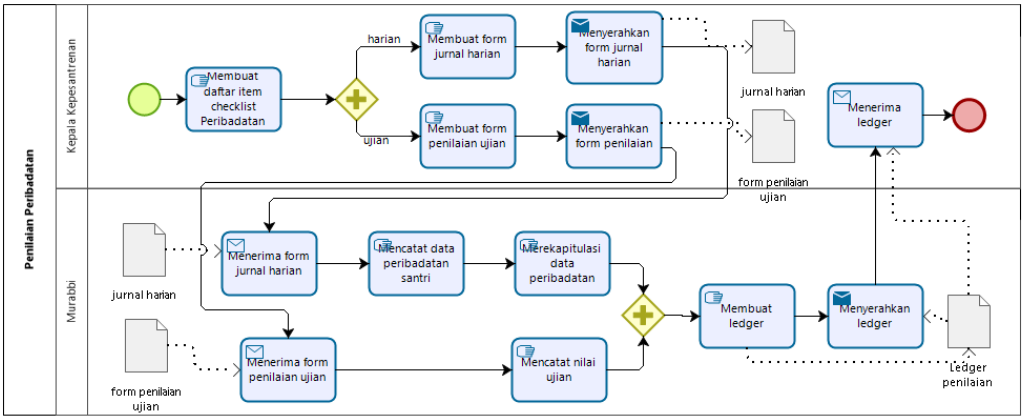


Gambar 4.2 Proses Bisnis *As-is* Penilaian Kemandirian

Gambar 4.2 menunjukkan proses bisnis *as-is* dari kegiatan penilaian kemandirian. Penilaian kemandirian adalah proses menilai kegiatan kemandirian yang dilakukan semua santri. Proses dimulai dari kepala kepesantrenan menentukan daftar item aktivitas apa saja yang akan dinilai. Setelah itu kepala kepesantrenan membuat lembar penilaian. Terdapat dua jenis *lembar* penilaian yaitu lembar jurnal harian dan lembar penilaian ujian. Lembar jurnal harian digunakan untuk menilai kuantitas aktivitas yang dilakukan santri. Sedangkan lembar penilaian ujian digunakan untuk menilai kualitas kemandirian santri. Lembar jurnal harian berisi daftar aktivitas yang harus dikerjakan santri setiap hari. Setiap hari aktivitas yang dinilai berbeda beda.

Setelah lembar jurnal harian dibuat, kepala kepesantrenan membagikan lembar tersebut untuk diisi para murabbi setiap hari. Isian dari lembar tersebut adalah apakah santri melaksanakan aktivitas atau tidak. Pada akhir semester, setiap murabbi merekapitulasi nilai yang sudah diisi. Selain itu terdapat lembar penilaian ujian yang digunakan untuk menilai kualitas kemandirian dari setiap santri. Penilaian ujian kemandirian biasa dilakukan di akhir semester. Setelah mendapatkan nilai harian dan ujian, murabbi membuat ledger dari kedua nilai tersebut. Setelah itu ledger diserahkan ke kepala kepesantrenan untuk laporan kegiatan kemandirian santri dalam satu semester.

4.1.1.3 Penilaian Peribadatan



Gambar 4.3 Proses Bisnis *As-is* Penilaian Peribadatan

Gambar 4.3 menunjukkan proses bisnis *as-is* dari kegiatan penilaian peribadatan. Proses penilaian peribadatan hampir sama dengan penilaian kemandirian. Proses dimulai dari kepala kepesantrenan membuat daftar item peribadatan yang dinilai oleh murabbi. Kemudian dibuat lembar penilaian untuk jurnal harian dan ujian. Setelah itu lembar disebarkan ke semua murabbi untuk diisi. Jurnal harian diisi setiap hari oleh murabbi. Sedangkan untuk mencatat nilai ujian dilakukan pada akhir semester setelah santri melaksanakan ujian praktik ibadah. Setelah didapatkan data harian dan ujian, murabbi membuat ledger peribadatan. Ledger yang sudah selesai dibuat diserahkan ke kepala kepesantrenan.

4.1.2 Analisis Permasalahan Proses Bisnis Saat Ini (*As-Is*)

Analisis permasalahan pada proses bisnis dilakukan guna menemukan permasalahan yang ada. Sehingga dapat ditentukan solusi terhadap permasalahan tersebut. Solusi yang ada digunakan sebagai dasar pembuatan proses bisnis rekomendasi (*to-be*). Adapun hasil dari analisis permasalahan akan dijelaskan pada Tabel 4.1.

Tabel 4.1 Analisis Permasalahan Proses Bisnis Saat Ini (*As-Is*)

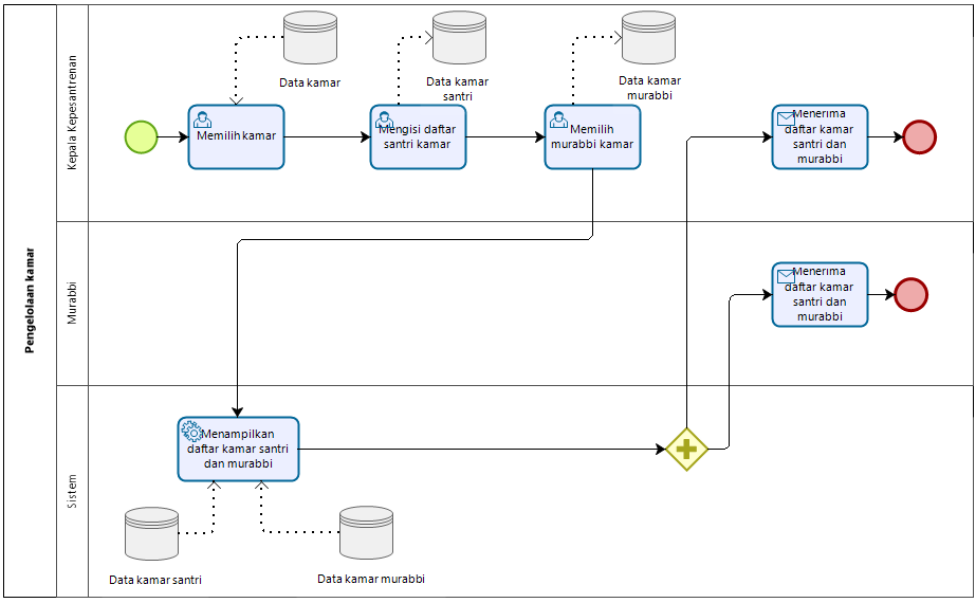
Permasalahan	Dampak	Solusi
Proses pencatatan aktivitas santri memerlukan banyak kertas	Membutuhkan dana yang banyak untuk pembuatan lembar penilaian dan memakan tempat untuk menyimpan hasil pencatatan	Membuat sistem informasi yang dapat menampung hasil pencatatan aktivitas santri

Proses pencatatan aktivitas santri oleh murabbi dilakukan secara manual	Murabbi harus bekerja dua kali dalam melakukan rekapitulasi nilai aktivitas santri secara manual dan terdapat risiko nilai kurang valid karena lembar pencatatan rusak atau hilang	Membuat sistem yang dapat digunakan untuk mencatat aktivitas santri sehingga hasil pencatatan dapat tersimpan lebih aman sekaligus melakukan rekapitulasi secara otomatis
Pembuatan ledger penilaian secara manual	Murabbi membutuhkan waktu yang cukup lama dalam membuat ledger penilaian	Membuat sistem yang dapat membantu dalam menghasilkan ledger dan melaporkannya secara otomatis
Banyaknya aktivitas yang dilakukan pada proses kepengasuhan	Memerlukan usaha dan waktu yang banyak dalam proses kepengasuhan	Menghilangkan aktivitas yang dirasa tidak perlu dan mengalihkan aktivitas yang dapat dialihkan ke sistem informasi

4.1.3 Proses Bisnis Usulan (*To-be*)

Pemodelan proses bisnis *to-be* merupakan pemodelan proses bisnis usulan sebagai bentuk penyelesaian masalah yang terdapat pada proses bisnis saat ini. Proses bisnis *to-be* tersebut digambarkan melalui model berikut:

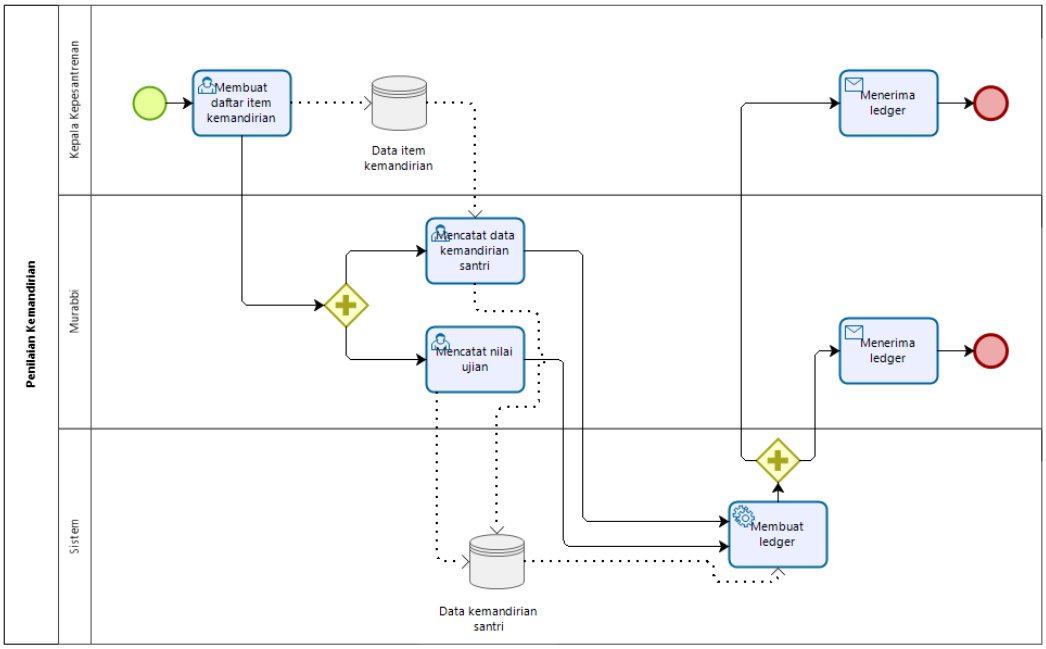
4.1.3.1 Pengelolaan Kamar



Gambar 4.4 Proses Bisnis *To Be* Pengelolaan Kamar

Gambar 4.4 menunjukkan proses bisnis *to-be* dari kegiatan pengelolaan kamar. Proses bisnis *to-be* dari pengelolaan kamar dimulai dari kepala kepesantrenan memilih kamar yang akan diisi. Selanjutnya kepala kepesantrenan menentukan siapa saja santri yang menempati kamar tersebut. Setelah itu menentukan murabbi dari kamar tersebut. Setelah semua hal tersebut dilakukan, maka sistem akan menyimpan daftar santri dan murabbi kamar. Proses berakhir setelah kepala kepesantrenan dan murabbi mendapatkan daftar kamar santri.

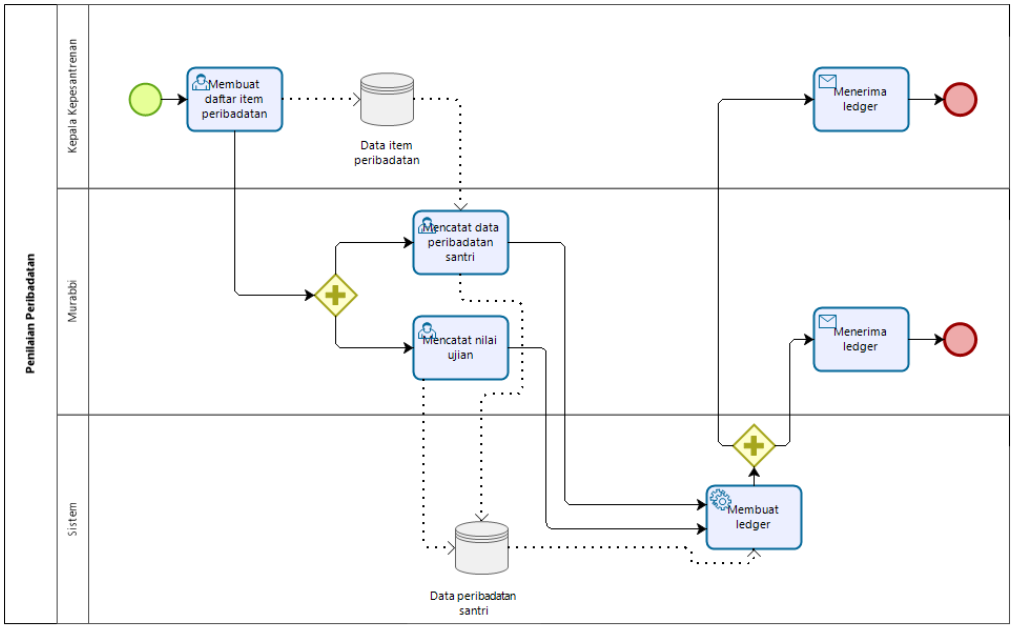
4.1.3.2 Penilaian Kemandirian



Gambar 4.5 Proses Bisnis *To Be* Penilaian Kemandirian

Gambar 4.5 menunjukkan proses bisnis *to-be* dari kegiatan penilaian kemandirian. Proses bisnis penilaian kemandirian dimulai dari kepala kepesantrenan membuat daftar item kemandirian. Daftar item tersebut dapat berbeda beda untuk masing-masing harinya. Kemudian murabbi melakukan pencatatan kemandirian secara rutin setiap hari sesuai item yang ada pada hari pengisian. Pencatatan nilai ujian dilakukan sekali pada waktu mendekati akhir semester. Setelah nilai harian dan nilai ujian sudah lengkap maka sistem akan membuat ledger secara otomatis. Setelah ledger dibuat, murabbi dan kepala kepesantrenan dapat melihatnya.

4.1.3.3 Penilaian Peribadatan



Gambar 4.6 Proses Bisnis *To Be* Penilaian Peribadatan

Gambar 4.6 menunjukkan proses bisnis *to-be* dari kegiatan penilaian peribadatan. Proses bisnis penilaian peribadatan dimulai dari kepala kepesantrenan membuat daftar item peribadatan. Daftar item peribadatan adalah item yang dinilai dari aktivitas ibadah harian. Item yang diisi setiap harinya sama. Kemudian murabbi melakukan pencatatan kemandirian secara rutin setiap hari. Pencatatan nilai ujian dilakukan sekali pada waktu mendekati akhir semester. Nilai ujian berasal dari hasil ujian praktik peribadatan santri. Setelah nilai harian dan nilai ujian sudah lengkap maka sistem akan membuat ledger secara otomatis. Setelah ledger dibuat, murabbi dan kepala kepesantrenan dapat melihatnya.

4.1.4 Analisis Proses Bisnis

Berdasarkan proses bisnis yang sudah digambarkan pada proses bisnis *to-be*, maka dapat digunakan sebagai acuan dalam membuat kebutuhan fungsional sistem informasi manajemen pengasuhan santri. Kebutuhan fungsional berasal dari aktivitas yang ada pada proses bisnis *to-be*. Aktivitas tersebut dapat dirincikan pada Tabel 4.2

Tabel 4.2 Daftar Proses Bisnis

Kode	Nama Proses Bisnis
PB01	Pengelolaan Kamar *To-Be*
PB02	Penilaian Kemandirian *To-Be*
PB03	Penilaian Peribadatan *To-Be*

Dari proses bisnis *to-be* selanjutnya dijelaskan apa saja aktivitas yang ada pada proses bisnis tersebut. Aktivitas-aktivitas tersebut nantinya digunakan sebagai acuan pembuatan kebutuhan fungsional. Aktivitas tersebut dijelaskan pada Tabel 4.3.

Tabel 4.3 Aktivitas Proses Bisnis

Kode Proses Bisnis	Kode Aktivitas	Nama	Deskripsi
PB01	AC01	Memilih kamar	Memilih kamar untuk diisi anggotanya
	AC02	Mengisi daftar santri kamar	Memilih santri untuk ditempatkan di kamar
	AC03	Memilih murabbi kamar	Memilih murabbi sebagai penanggung jawab kamar
	AC04	Menampilkan daftar kamar santri dan murabbi	Sistem menampilkan daftar santri dan murabbi yang sudah ditambahkan
	AC05	Menerima daftar kamar santri dan murabbi	Melihat daftar anggota kamar dan murabbi yang bertanggungjawab
PB02	AC06	Membuat daftar item kemandirian	Mengelola daftar aktivitas yang dinilai dalam kemandirian
	AC07	Mencatat data kemandirian santri	Melakukan pencatatan data harian kemandirian santri
	AC08	Mencatat nilai ujian	Melakukan pencatatan hasil ujian praktik kemandirian santri
	AC09	Membuat ledger	Membuat ledger penilaian secara otomatis dari hasil penilaian
	AC10	Melihat ledger	Kepala kepesantrenan

			dan murabbi dapat melihat ledger secara otomatis
PB03	AC11	Membuat daftar item peribadatan	Mengelola daftar aktivitas yang dinilai dalam proses peribadatan harian santri
	AC12	Mencatat data peribadatan santri	Melakukan pencatatan data harian peribadatan santri
	AC13	Mencatat nilai ujian	Melakukan pencatatan hasil ujian praktik ibadah santri
	AC14	Membuat ledger	Membuat ledger penilaian secara otomatis dari hasil penilaian
	AC15	Melihat ledger	Kepala kepesantrenan dan murabbi dapat melihat ledger secara otomatis

4.2 Analisis Kebutuhan

Analisis kebutuhan sistem dilakukan untuk menentukan kebutuhan fungsional sistem. Kebutuhan tersebut berdasarkan hasil analisis proses bisnis sebelumnya seperti yang telah dijelaskan padaTabel 4.3. Pada proses analisis kebutuhan juga dilakukan pembuatan diagram dan *use case scenario*.

4.2.1 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan atau fungsi yang harus disediakan oleh sistem informasi manajemen pengasuh. Tabel 4.4 menunjukkan kebutuhan fungsional yang dimiliki sistem informasi manajemen pengasuh.

Tabel 4.4 Kebutuhan Fungsional Sistem

Kode Aktivitas	Kode Fungsional	Nama Fungsional	Deskripsi
-	KF-01	*Login*	Sistem dapat melakukan autentikasi dan membagi hak akses

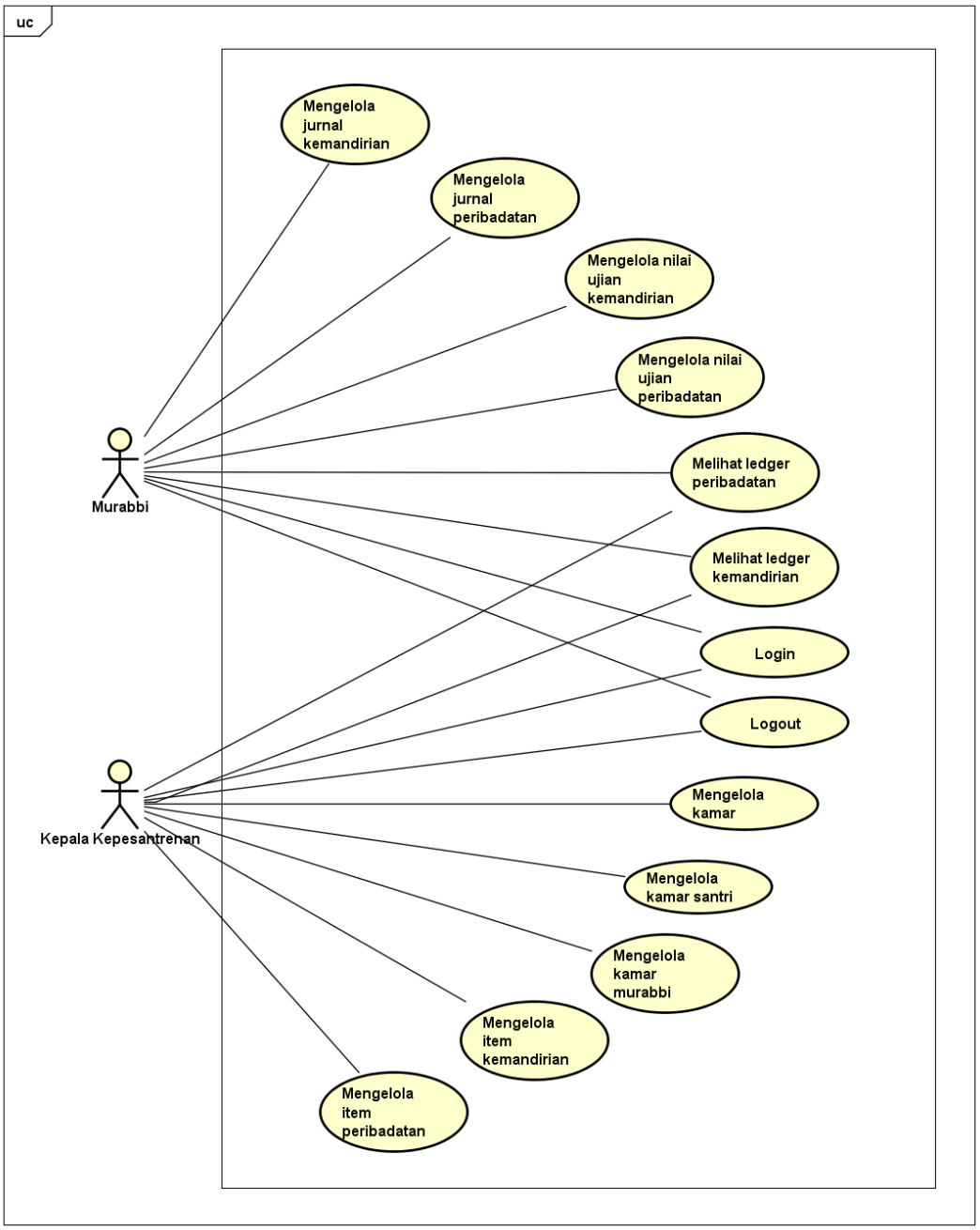
			kepada pengguna sistem
AC01	KF-02	Melihat data kamar	Sistem dapat menampilkan informasi kamar yang ada
	KF-03	Mengubah data kamar	Sistem dapat digunakan untuk mengubah informasi kamar
	KF-04	Mencari kamar	Sistem dapat digunakan untuk mencari kamar
AC02	KF-05	Menambah anggota kamar	Sistem dapat digunakan untuk menambahkan anggota kamar
	KF-06	Menghapus anggota kamar	Sistem dapat digunakan untuk menghapus anggota kamar
AC03	KF-07	Menambah murabbi kamar	Sistem dapat digunakan untuk menambahkan murabbi kamar
	KF-08	Menghapus murabbi kamar	Sistem dapat digunakan untuk menghapus murabbi kamar
	KF-09	Mengubah murabbi kamar	Sistem dapat digunakan untuk mengubah murabbi kamar
AC04	KF-10	Melihat daftar anggota kamar	Sistem dapat digunakan untuk melihat informasi anggota kamar
	KF-11	Melihat murabbi kamar	Sistem dapat digunakan untuk melihat informasi murabbi kamar
AC06	KF-12	Menambah item kemandirian	Sistem dapat digunakan untuk menambah item aktivitas kemandirian
	KF-13	Mengubah item kemandirian	Sistem dapat digunakan untuk mengubah informasi

			item aktivitas kemandirian
	KF-14	Menghapus item kemandirian	Sistem dapat digunakan untuk menghapus item aktivitas kemandirian
	KF-15	Melihat item kemandirian	Sistem dapat digunakan untuk melihat daftar item aktivitas kemandirian
AC07	KF-16	Menambah jurnal kemandirian	Sistem dapat digunakan untuk menambah data jurnal kemandirian santri
	KF-17	Menghapus jurnal kemandirian	Sistem dapat digunakan untuk menghapus data jurnal kemandirian
	KF-18	Mengubah jurnal kemandirian	Sistem dapat digunakan untuk mengubah data jurnal
	KF-19	Menampilkan jurnal kemandirian	Sistem dapat menampilkan informasi jurnal yang sudah dibuat
AC08	KF-20	Menambah nilai ujian kemandirian	Sistem dapat digunakan untuk menambah data nilai ujian kemandirian
	KF-21	Mengubah nilai ujian kemandirian	Sistem dapat digunakan untuk mengubah data nilai ujian kemandirian
AC09,AC10	KF-22	Menampilkan ledger kemandirian	Sistem dapat menampilkan ledger kemandirian
AC11	KF-23	Menambah item peribadatan	Sistem dapat digunakan untuk menambah item aktivitas peribadatan
	KF-24	Mengubah item peribadatan	Sistem dapat digunakan untuk mengubah informasi item aktivitas peribadatan
	KF-25	Menghapus item peribadatan	Sistem dapat digunakan untuk

			menghapus item aktivitas peribadatan
	KF-26	Melihat item peribadatan	Sistem dapat digunakan untuk melihat daftar item aktivitas peribadatan
AC12	KF-27	Menambah jurnal peribadatan	Sistem dapat digunakan untuk menambah data jurnal peribadatan santri
	KF-28	Menghapus jurnal peribadatan	Sistem dapat digunakan untuk menghapus data jurnal peribadatan
	KF-29	Mengubah jurnal peribadatan	Sistem dapat digunakan untuk mengubah data jurnal
	KF-30	Menampilkan jurnal peribadatan	Sistem dapat menampilkan informasi jurnal yang sudah dibuat
AC13	KF-31	Menambah nilai ujian peribadatan	Sistem dapat digunakan untuk menambah data nilai ujian peribadatan
	KF-32	Mengubah nilai ujian peribadatan	Sistem dapat digunakan untuk mengubah data nilai ujian peribadatan
AC14, AC15	KF-33	Menampilkan ledger peribadatan	Sistem dapat menampilkan ledger peribadatan

4.2.2 Pemodelan Diagram *Use Case*

Pemodelan diagram *use case* ditujukan untuk menggambarkan secara ringkas siapa yang menggunakan sistem dan apa saja yang bisa dilakukannya. Pada Gambar 4.7 menunjukkan *use case* pada sistem informasi manajemen pengasuhan santri yang menggambarkan aktor dan interaksinya dengan sistem.



Gambar 4.7 *Use case Diagram*

Terdapat dua aktor yang terlibat dengan sistem yaitu kepala kepesantrenan dan murabbi. Tabel 4.5 menjelaskan deskripsi dari setiap aktor dan bagaimana aktivitas yang dilakukan.

Tabel 4.5 Deskripsi Aktor

No.	Aktor	Deskripsi
1	Kepala kepesantrenan	Aktor kepala kepesantrenan merupakan pengguna sistem yang memiliki *use case login*, mengelola kamar, mengelola kamar santri, mengelola kamar murabbi, melihat ledger kemandirian dan melihat ledger peribadatan
2	Murabbi	Aktor murabbi merupakan pengguna sistem yang memiliki *use case login*, mengelola kamar, mengelola jurnal kemandirian, mengelola jurnal peribadatan, mengelola nilai ujian kemandirian, mengelola nilai ujian peribadatan, melihat ledger kemandirian dan peribadatan

4.2.3 Use Case Scenario

Setelah berhasil mendefinisikan *use case,* tahap selanjutnya adalah masing-masing dari *use case* tersebut akan dijelaskan lebih detail ke dalam *use case scenario*. *Use case scenario* adalah salah satu dari diagram UML yang bertujuan untuk memberikan gambaran umum tentang fungsionalitas suatu proses bisnis yang di dalamnya melibatkan sebuah sistem.

4.2.3.1 *Use case scenario: login*

Pada Tabel 4.6menunjukkan skenario dari *use case login*. Skenario tersebut menggambarkan bagaimana aktor dapat masuk ke sistem dan mendapatkan hak akses. Aliran dasar *use case login* dimulai dari aktor mengisi *username* dan *password* sampai aktor berhasil mendapatkan hak akses dan membuka halaman utama sistem. Selain itu terdapat alternatif ketika *username* atau *password* yang dimasukkan salah.

Tabel 4.6 *Use Case Scenario*: *Login*

Nama	*Login*
Brief description	*Use case* ini menggambarkan bagaimana aktor dapat masuk ke sistem dan mendapatkan hak akses
Aktor	Kepala kepesantrenan dan murabbi
Precondition	• Aktor sudah memiliki *username* dan *password* • Aktor sudah mengakses halaman *login*
Basic flow	**{Mengisi username dan password}** 1. *Use case* dimulai ketika aktor telah mengakses halaman login 2. *Aktor* mengisi *username* dan *password* 3. Aktor menjalankan fungsi login **{Autentikasi aktor}** 4. Sistem melakukan autentikasi *username* dan *password*

	5. Sistem memberikan *session* sebagai identifikasi aktor dan hak akses **{*Use case* selesai}** 6. Sistem menampilkan halaman utama 7. *Use case* berakhir
Alternative Flow	A. Salah mengisi password dan/atau username 1. Pada saat **{mengisi username dan password}**, jika aktor salah memasukkan username atau password 2. Sistem menolak autentikasi pengguna 3. Sistem menampilkan halaman *login*
Postconditions	• Sistem menampilkan halaman utama • Sistem menampilkan halaman *login*

4.2.3.2 *Use case scenario: logout*

Pada Tabel 4.7 menunjukkan skenario dari *use case logout*. Skenario tersebut menggambarkan bagaimana aktor keluar dari sistem. Aliran dasar *use case logout* dimulai ketika aktor memilih fungsi *logout*. Hasil akhir dari skenario ini adalah sistem menampilkan halaman *logout*.

Tabel 4.7 *Use Case Scenario*: *Logout*

Nama	*Logout*
Brief description	*Use case* ini menggambarkan bagaimana aktor keluar dari sistem
Aktor	Kepala kepesantrenan dan murabbi
Precondition	• Aktor sudah masuk ke dalam sistem
Basic flow	**{menjalankan logout}** 1. *Use case* dimulai ketika aktor memilih fungsi *logout* **{*Use case* selesai}** 2. Sistem menampilkan halaman *login* 3. *Use case* berakhir
Alternative Flow	-
Postconditions	• Sistem menampilkan halaman *login*

4.2.3.3 *Use case scenario*: mengelola kamar

Pada Tabel 4.8 menunjukkan skenario dari *use case* mengelola kamar. Skenario tersebut menggambarkan bagaimana aktor mengelola data kamar. Aliran dasar *use case* mengelola kamar dimulai ketika sistem menampilkan halaman kelola kamar. Hasil akhir dari skenario ini adalah aktor berhasil menambahkan kamar dan menampilkan data kamar yang ada.

Tabel 4.8 *Use Case Scenario*: Mengelola Kamar

Nama	Mengelola kamar
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola kamar
Aktor	Kepala kepesantrenan
Precondition	• Kepala kepesantrenan telah menjalankan *use case login* • Kepala kepesantrenan memilih menu kelola kamar
Basic flow	**{Menampilkan halaman kelola kamar}** 1. *Use case* dimulai ketika sistem menampilkan halaman menu kelola kamar dan fungsi yang tersedia **{Menambah kamar}** 2. Aktor memilih fungsi tambah kamar 3. Sistem meminta aktor mengisi data kamar 4. Kepala kepesantrenan mengisi data kamar 5. Sistem menyimpan informasi kamar yang dimasukkan aktor **{*Use case* selesai}** 6. Sistem menampilkan daftar kamar dan fungsi yang tersedia 7. *Use case* berakhir
Alternative Flow	A. Menyunting kamar 1. Pada saat **{menampilkan daftar kamar}**, kepala kepesantrenan memilih fungsi sunting kamar yang ada pada daftar kamar 2. Sistem meminta aktor mengubah data kamar yang dipilih 3. Kepala kepesantrenan mengubah data kamar 4. Sistem memperbarui data kamar yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}** B. Mencari kamar 1. Pada saat **{menampilkan daftar kamar}**, kepala kepesantrenan memilih fungsi cari kamar yang ada pada halaman daftar kamar 2. Aktor mengirim parameter kamar yang akan dicari 3. Sistem menampilkan daftar kamar yang sesuai dengan parameter yang dicari 4. *Use case* berakhir
Postconditions	Sistem menampilkan daftar kamar dan fungsi yang tersedia

4.2.3.4 *Use case scenario*: mengelola kamar santri

Pada Tabel 4.9 menunjukkan skenario dari *use case* mengelola kamar santri. Skenario tersebut menggambarkan bagaimana aktor mengelola anggota kamar.

Aliran dasar *use case* mengelola kamar santri dimulai ketika sistem menampilkan halaman kelola kamar santri. Terdapat aliran alternatif ketika aktor ingin menambahkan anggota kamar dan menghapus anggota kamar.

Tabel 4.9 *Use Case Scenario*: Mengelola Kamar Santri

Nama	Mengelola kamar santri
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola anggota kamar
Aktor	Kepala kepesantrenan
Precondition	• Kepala kepesantrenan telah menjalankan *use case login* • Data santri sudah ada dalam sistem • Kepala kepesantrenan memilih menu kelola kamar santri
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar dan fungsi yang tersedia **{Memilih kamar}** 2. Kepala kepesantrenan memilih kamar 3. Sistem menampilkan informasi kamar, daftar anggota dan menu yang tersedia **{*Use case* selesai}** 4. Sistem menampilkan informasi kamar, daftar anggota dan menu yang tersedia 5. *Use case* berakhir
Alternative Flow	A. Menambah anggota 1. Pada saat **{memilih kamar},** kepala kepesantrenan memilih fungsi tambah anggota kamar 2. Sistem meminta aktor untuk memilih anggota kamar 3. Kepala kepesantrenan memilih anggota kamar 4. Sistem menyimpan data yang dimasukkan aktor B. Menghapus anggota 1. Pada saat **{memilih kamar}**, kepala kepesantrenan memilih fungsi hapus anggota yang ada pada daftar anggota 2. Sistem mengonfirmasi untuk menghapus anggota 3. Kepala kepesantrenan menyetujui untuk hapus anggota 4. Sistem menghapus anggota yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan informasi kamar, daftar anggota dan menu yang tersedia

4.2.3.5 *Use case scenario*: mengelola murabbi kamar

Pada Tabel 4.10 menunjukkan skenario dari *use case* mengelola murabbi kamar. Skenario tersebut menggambarkan bagaimana aktor mengelola murabbi suatu kamar kamar. Aliran dasar *use case* mengelola murabbi kamar dimulai ketika sistem menampilkan halaman kelola murabbi kamar. Terdapat aliran alternatif ketika aktor ingin menyunting dan menghapus murabbi kamar.

Tabel 4.10 *Use Case Scenario*: Mengelola Murabbi Kamar

Nama	Mengelola murabbi kamar
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola murabbi kamar
Aktor	Kepala kepesantrenan
Precondition	• Kepala kepesantrenan telah menjalankan *use case login* • Kepala kepesantrenan memilih menu kelola murabbi kamar
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar beserta murabbi dan fungsi yang tersedia **{Menambah murabbi kamar}** 2. Kepala kepesantrenan menjalankan fungsi tambah murabbi kamar 3. Sistem meminta aktor untuk memasukkan murabbi kamar 4. Kepala kepesantrenan mengisi murabbi kamar 5. Sistem menyimpan data yang dimasukkan kepala kepesantrenan **{*Use case* selesai}** 6. Sistem menampilkan daftar kamar beserta murabbi dan fungsi yang tersedia 7. *Use case* berakhir
Alternative Flow	A. Menyunting murabbi kamar 1. Pada saat **{menampilkan daftar kamar}**, kepala kepesantrenan memilih kamar yang akan disunting 2. Sistem meminta aktor untuk mengubah murabbi kamar 3. Kepala kepesantrenan memilih murabbi kamar 4. Sistem menyimpan data yang dimasukkan aktor 5. Kembali ke *basic flow* **{*use case* selesai}** B. Menghapus murabbi kamar 1. Pada saat **{menampilkan daftar kamar}**, kepala kepesantrenan memilih kamar yang akan dihapus murabbinya 2. Sistem mengonfirmasi untuk menghapus murabbi kamar

	3. Kepala kepesantrenan menyetujui untuk menghapus murabbi kamar 4. Sistem menghapus murabbi kamar yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan daftar kamar beserta murabbi dan fungsi yang tersedia

4.2.3.6 *Use case scenario*: mengelola item kemandirian

Pada Tabel 4.11 menunjukkan skenario dari *use case* mengelola item kemandirian. Skenario tersebut menggambarkan bagaimana aktor mengelola item/aspek yang dinilai pada jurnal kemandirian. Terdapat aliran alternatif ketika aktor ingin menyunting dan menghapus item kemandirian.

Tabel 4.11 *Use Case Scenario*: Mengelola Item Kemandirian

Nama	Mengelola item kemandirian
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola item yang dinilai pada jurnal kemandirian
Aktor	Kepala kepesantrenan
Precondition	• Kepala kepesantrenan telah menjalankan *use case login* • Kepala kepesantrenan membuka halaman kelola item kemandirian
Basic flow	**{Menampilkan daftar item}** 1. *Use case* dimulai ketika sistem menampilkan daftar item kemandirian dan fungsi yang tersedia **{Menambah item kemandirian}** 2. Kepala kepesantrenan memilih fungsi tambah item 3. Sistem meminta aktor untuk memasukkan data item kemandirian 4. Kepala kepesantrenan mengisi data kemandirian 5. Sistem menyimpan data yang dimasukkan kepala kepesantrenan **{*Use case* selesai}** 6. Sistem menampilkan daftar item dan fungsi yang tersedia 7. *Use case* berakhir
Alternative Flow	A. Menyunting item kemandirian 1. Pada saat **{menampilkan daftar item}**, kepala kepesantrenan memilih item yang akan disunting 2. Sistem meminta aktor untuk mengubah data item 3. Kepala kepesantrenan memasukkan data item 4. Sistem menyimpan data yang dimasukkan aktor 5. Kembali ke *basic flow* **{*use case* selesai}** B. Menghapus item kemandirian

	1. Pada saat **{menampilkan daftar item}**, kepala kepesantrenan memilih item yang akan dihapus 2. Sistem mengonfirmasi untuk menghapus item 3. Kepala kepesantrenan menyetujui untuk menghapus item 4. Sistem menghapus item yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan daftar item dan fungsi yang tersedia

4.2.3.7 *Use case scenario*: mengelola item peribadatan

Pada Tabel 4.12 menunjukkan skenario dari *use case* mengelola item peribadatan. Skenario tersebut menggambarkan bagaimana aktor mengelola item/aspek yang dinilai pada jurnal peribadatan. Terdapat aliran alternatif ketika aktor ingin menyunting dan menghapus item peribadatan.

Tabel 4.12 *Use Case Scenario*: Mengelola Item Peribadatan

Nama	Mengelola item peribadatan
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola item yang dinilai pada jurnal peribadatan
Aktor	Kepala kepesantrenan
Precondition	• Kepala kepesantrenan telah menjalankan *use case login* • Kepala kepesantrenan membuka halaman kelola item peribadatan
Basic flow	**{Menampilkan daftar item}** 1. *Use case* dimulai ketika sistem menampilkan daftar item peribadatan dan fungsi yang tersedia **{Menambah item peribadatan}** 2. Kepala kepesantrenan menjalankan fungsi tambah item 3. Sistem meminta aktor untuk memasukkan data item peribadatan 4. Kepala kepesantrenan mengisi data item peribadatan 5. Sistem menyimpan data yang dimasukkan kepala kepesantrenan **{*Use case* selesai}** 6. Sistem menampilkan daftar item dan fungsi yang tersedia 7. *Use case* berakhir
Alternative Flow	A. Menyunting item peribadatan 1. Pada saat **{menampilkan daftar item}**, kepala kepesantrenan memilih item yang akan disunting 2. Sistem meminta aktor untuk mengubah data item 3. Kepala kepesantrenan memasukkan data item

	4. Sistem menyimpan data yang dimasukkan aktor 5. Kembali ke *basic flow* **{*use case* selesai}** B. Menghapus item peribadatan 1. Pada saat **{menampilkan daftar item}**, kepala kepesantrenan memilih item yang akan dihapus 2. Sistem mengonfirmasi untuk menghapus item 3. Kepala kepesantrenan menyetujui untuk menghapus item 4. Sistem menghapus item yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan daftar item dan fungsi yang tersedia

4.2.3.8 *Use case scenario*: mengelola jurnal kemandirian

Pada Tabel 4.13 menunjukkan skenario dari *use case* mengelola jurnal kemandirian. Skenario tersebut menggambarkan bagaimana aktor mengelola jurnal kemandirian. Aliran dasar *use case* mengelola jurnal kemandirian adalah menambah jurnal kemandirian. Terdapat aliran alternatif ketika aktor ingin menyunting dan menghapus jurnal kemandirian.

Tabel 4.13 *Use Case Scenario*: Mengelola Jurnal Kemandirian

Nama	Mengelola jurnal kemandirian
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola jurnal kemandirian
Aktor	Murabbi
Precondition	• Murabbi telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar • Data item aktivitas kemandirian sudah ada • Murabbi berhasil membuka halaman kelola jurnal kemandirian
Basic flow	**{Menampilkan daftar jurnal}** 1. *Use case* dimulai ketika sistem menampilkan daftar jurnal yang sudah dibuat dan fungsi yang tersedia **{tambah jurnal}** 2. Murabbi menjalankan fungsi tambah jurnal **{Memilih kamar dan tanggal}** 3. Sistem meminta murabbi untuk memilih kamar dan tanggal yang akan diisi 4. Murabbi memilih kamar dan tanggal **{Mengisi jurnal}** 5. Sistem menampilkan daftar anggota kamar untuk diisi 6. Murabbi mengisi keterlaksanaan kemandirian santri 7. Sistem menyimpan data yang dimasukkan murabbi **{*Use case* selesai}**

	8. Sistem menampilkan halaman utama kelola jurnal kemandirian 9. *Use case* berakhir
Alternative Flow	A. Menyunting jurnal 1. Pada saat **{menampilkan daftar jurnal}**, murabbi memilih jurnal yang akan disunting 2. Sistem meminta murabbi untuk mengubah data jurnal 3. Murabbi mengubah data jurnal 4. Sistem menyimpan data yang dimasukkan murabbi 5. Kembali ke *basic flow* **{*use case* selesai}** B. Menghapus jurnal 1. Pada saat **{menampilkan daftar jurnal}**, murabbi memilih jurnal yang akan dihapus 2. Sistem mengonfirmasi untuk menghapus jurnal 3. Murabbi menyetujui untuk menghapus jurnal 4. Sistem menghapus jurnal yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan halaman utama kelola jurnal kemandirian

4.2.3.9 *Use case scenario*: mengelola jurnal peribadatan

Pada Tabel 4.14 menunjukkan skenario dari *use case* mengelola jurnal peribadatan. Skenario tersebut menggambarkan bagaimana aktor mengelola jurnal peribadatan. Aliran dasar *use case* mengelola jurnal peribadatan adalah menambah jurnal peribadatan. Terdapat aliran alternatif ketika aktor ingin menyunting dan menghapus jurnal peribadatan.

Tabel 4.14 *Use Case Scenario*: Mengelola Jurnal Peribadatan

Nama	Mengelola jurnal peribadatan
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola jurnal peribadatan
Aktor	Murabbi
Precondition	• Murabbi telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar • Data item aktivitas peribadatan sudah ada • Murabbi berhasil membuka halaman kelola jurnal peribadatan
Basic flow	**{Menampilkan daftar jurnal}** 1. *Use case* dimulai ketika sistem menampilkan daftar jurnal yang sudah dibuat dan fungsi yang tersedia **{tambah jurnal}** 2. Murabbi menjalankan fungsi tambah jurnal

	{Memilih kamar dan tanggal} 3. Sistem meminta murabbi untuk memilih kamar dan tanggal yang akan diisi 4. Murabbi memilih kamar dan tanggal **{Mengisi jurnal}** 5. Sistem menampilkan daftar anggota kamar untuk diisi 6. Murabbi mengisi keterlaksanaan peribadatan santri 7. Sistem menyimpan data yang dimasukkan murabbi **{*Use case* selesai}** 8. Sistem menampilkan halaman utama kelola jurnal peribadatan 9. *Use case* berakhir
Alternative Flow	A. Menyunting jurnal 1. Pada saat **{menampilkan daftar jurnal}**, murabbi memilih jurnal yang akan disunting 2. Sistem meminta murabbi untuk mengubah data jurnal 3. Murabbi mengubah data jurnal 4. Sistem menyimpan data yang dimasukkan murabbi 5. Kembali ke *basic flow* **{*use case* selesai}** B. Menghapus jurnal 1. Pada saat **{menampilkan daftar jurnal}**, murabbi memilih jurnal yang akan dihapus 2. Sistem mengonfirmasi untuk menghapus jurnal 3. Murabbi menyetujui untuk menghapus jurnal 4. Sistem menghapus jurnal yang dipilih 5. Kembali ke *basic flow* **{*use case* selesai}**
Postconditions	Sistem menampilkan halaman utama kelola jurnal peribadatan

4.2.3.10 *Use case scenario*: mengelola nilai ujian kemandirian

Pada Tabel 4.15 menunjukkan skenario dari *use case* mengelola nilai ujian kemandirian. Skenario tersebut menggambarkan bagaimana aktor mengelola nilai ujian kemandirian. Aliran dasar *use case* mengelola nilai ujian kemandirian adalah mengisi nilai ujian kemandrian.

Tabel 4.15 *Use Case Scenario*: Mengelola Nilai Ujian Kemandirian

Nama	Mengelola nilai ujian kemandirian
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola nilai ujian kemandirian
Aktor	Murabbi
Precondition	• Murabbi telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar

	• Murabbi berhasil membuka halaman kelola nilai ujian kemandirian
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar yang diasuh dan fungsi yang tersedia **{Mengisi nilai}** 2. Murabbi memilih kamar yang akan diisi nilainya 3. Sistem menampilkan daftar anggota kamar untuk diisi 4. Murabbi mengisi nilai ujian kemandirian santri 5. Sistem menyimpan data yang dimasukkan murabbi **{*Use case* selesai}** 6. Sistem menampilkan halaman utama kelola nilai ujian kemandirian 7. *Use case* berakhir
Alternative Flow	-
Postconditions	Sistem menampilkan halaman utama kelola nilai ujian kemandirian

4.2.3.11 *Use case scenario*: mengelola nilai ujian peribadatan

Pada Tabel 4.16 menunjukkan skenario dari *use case* mengelola nilai ujian peribadatan. Skenario tersebut menggambarkan bagaimana aktor mengelola nilai ujian peribadatan. Aliran dasar *use case* mengelola nilai ujian peribadatan adalah mengisi nilai ujian peribadatan.

Tabel 4.16 *Use Case Scenario*: Mengelola Nilai Ujian Peribadatan

Nama	Mengelola nilai ujian peribadatan
Brief description	*Use case* ini menggambarkan bagaimana aktor mengelola nilai ujian peribadatan
Aktor	Murabbi
Precondition	• Murabbi telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar • Murabbi berhasil membuka halaman kelola nilai ujian peribadatan
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar yang diasuh dan fungsi yang tersedia **{Mengisi nilai}** 2. Murabbi memilih kamar yang akan diisi nilainya 3. Sistem menampilkan daftar anggota kamar untuk diisi 4. Murabbi mengisi nilai ujian peribadatan santri 5. Sistem menyimpan data yang dimasukkan murabbi **{*Use case* selesai}**

	6. Sistem menampilkan halaman utama kelola nilai ujian peribadatan 7. *Use case* berakhir
Alternative Flow	-
Postconditions	Sistem menampilkan halaman utama kelola nilai ujian peribadatan

4.2.3.12 *Use case scenario*: melihat ledger kemandirian

Pada Tabel 4.17 menunjukkan skenario dari *use case* melihat ledger kemandirian. Skenario tersebut menggambarkan bagaimana aktor melihat ledger kemandirian. Aliran dasar *use case* melihat ledger kemandirian dimulai dari sistem menampilkan daftar kamar sampai sistem menampilkan informasi nilai kemandirian.

Tabel 4.17 *Use Case Scenario*: Melihat Ledger Kemandirian

Nama	Melihat ledger kemandirian
Brief description	*Use case* ini menggambarkan bagaimana aktor melihat ledger nilai kemandirian
Aktor	Murabbi dan Kepala Kepesantrenan
Precondition	• Aktor telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar • Aktor berhasil membuka halaman ledger kemandirian
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar dan fungsi yang tersedia **{Melihat ledger}** 2. Aktor memilih kamar yang akan dilihat nilainya 3. Sistem menampilkan informasi nilai anggota kamar dan status ketuntasannya **{*Use case* selesai}** 4. *Use case* berakhir
Alternative Flow	A. Tidak ada nilai jurnal harian atau nilai ujian 1. Pada saat **{melihat ledger }**, aktor memilih kamar yang nilainya belum memiliki nilai jurnal atau ujian 2. Sistem menampilkan nilai anggota kamar tanpa menampilkan status ketuntasan 3. *Use case* berakhir
Postconditions	• Sistem menampilkan informasi nilai anggota kamar dan status ketuntasannya • Sistem menampilkan nilai anggota kamar tanpa menampilkan status ketuntasan

4.2.3.13 *Use case scenario*: melihat ledger peribadatan

Pada Tabel 4.18 menunjukkan skenario dari *use case* melihat ledger peribadatan. Skenario tersebut menggambarkan bagaimana aktor melihat ledger peribadatan. Aliran dasar *use case* melihat ledger peribadatan dimulai dari sistem menampilkan daftar kamar sampai sistem menampilkan informasi nilai peribadatan.

Tabel 4.18 *Use Case Scenario*: Melihat Ledger Peribadatan

Nama	Melihat ledger peribadatan
Brief description	*Use case* ini menggambarkan bagaimana aktor melihat ledger nilai peribadatan
Aktor	Murabbi dan Kepala Kepesantrenan
Precondition	• Aktor telah menjalankan *use case login* • Murabbi telah memiliki kamar untuk diasuh • Kamar yang diasuh sudah memiliki anggota kamar • Aktor berhasil membuka halaman ledger peribadatan
Basic flow	**{Menampilkan daftar kamar}** 1. *Use case* dimulai ketika sistem menampilkan daftar kamar dan fungsi yang tersedia **{Melihat ledger}** 2. Aktor memilih kamar yang akan dilihat nilainya 3. Sistem menampilkan informasi nilai anggota kamar dan status ketuntasannya **{*Use case* selesai}** 4. *Use case* berakhir
Alternative Flow	A. Tidak ada nilai jurnal harian atau nilai ujian 1. Pada saat **{melihat ledger }**, aktor memilih kamar yang nilainya belum memiliki nilai jurnal atau ujian 2. Sistem menampilkan nilai anggota kamar tanpa menampilkan status ketuntasan 3. *Use case* berakhir
Postconditions	• Sistem menampilkan informasi nilai anggota kamar dan status ketuntasannya • Sistem menampilkan nilai anggota kamar tanpa menampilkan status ketuntasan

4.3 Perancangan Sistem

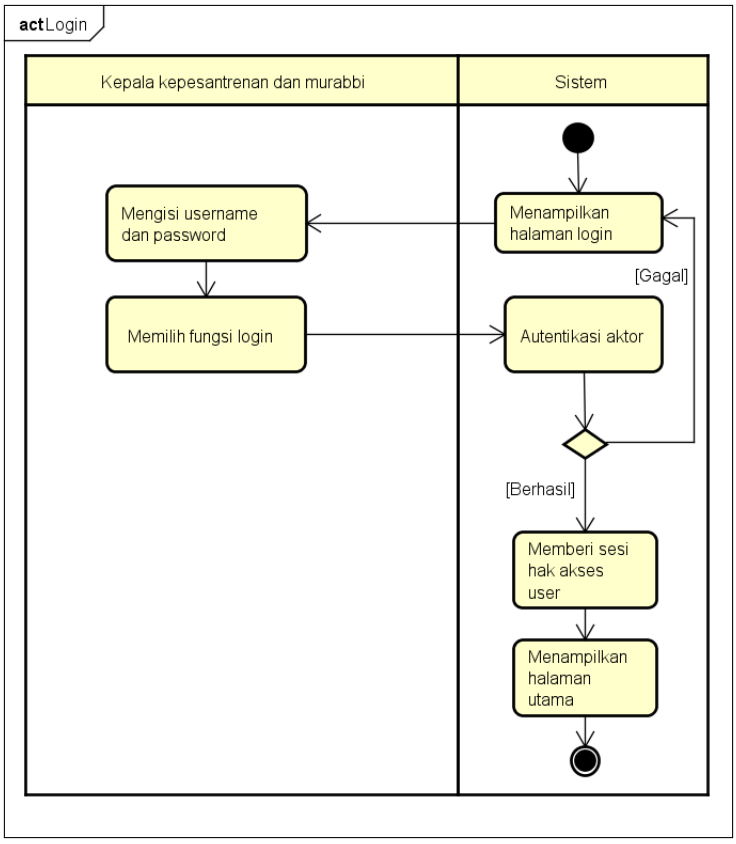
Setelah melakukan tahapan analisis kebutuhan, tahapan selanjutnya adalah perancangan sistem. Seluruh perancangan sistem yang dibuat didasarkan pada hasil analisis kebutuhan. Perancangan sistem dimaksudkan untuk merepresentasikan sistem yang dibangun. Selain itu perancangan digunakan sebagai dasar mengimplementasikan sistem ke dalam bahasa pemrograman.

4.3.1 Activity Diagram

Activity Diagram berfungsi memperlihatkan urutan aktivitas proses pada Sistem Informasi Manajemen Kepengasuhan. *Activity diagram* membantu memahami proses secara keseluruhan. *Activity diagram* dibuat berdasarkan *use case* yang telah dijelaskan sebelumnya.

4.3.1.1 Activity Diagram Login

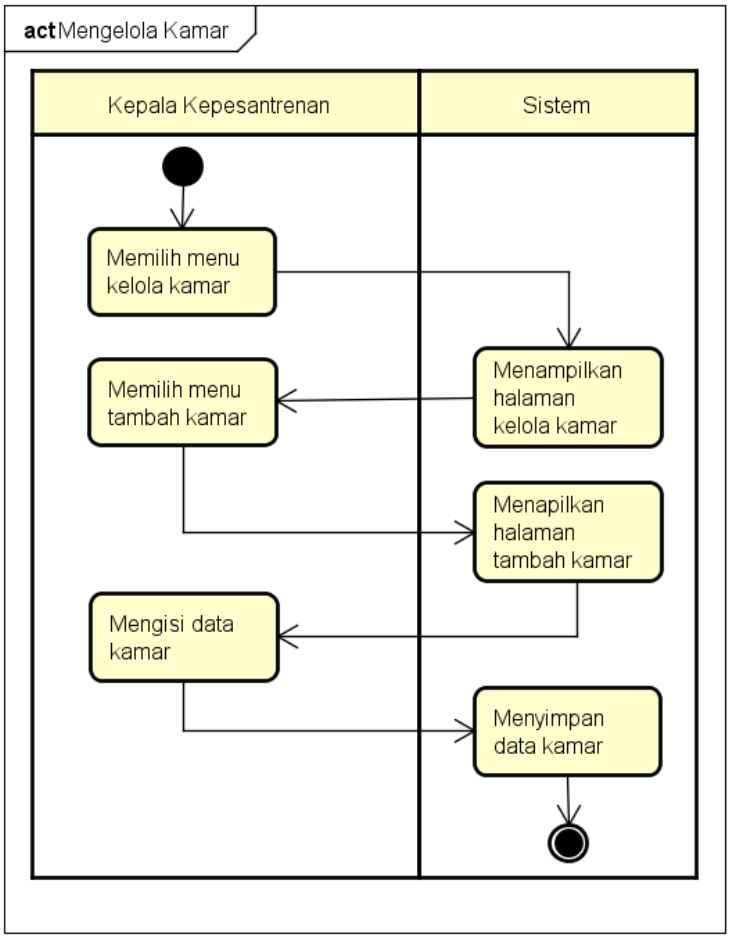
Gambar 4.8 menunjukkan diagram aktivitas pada proses *login*. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario login* pada Tabel 4.6. Aktivitas dimulai dari sistem menampilkan halaman *login* sampai aktor berhasil masuk ke dalam sistem.



Gambar 4.8 *Activity Diagram Login*

4.3.1.2 *Activity Diagram* Mengelola Kamar

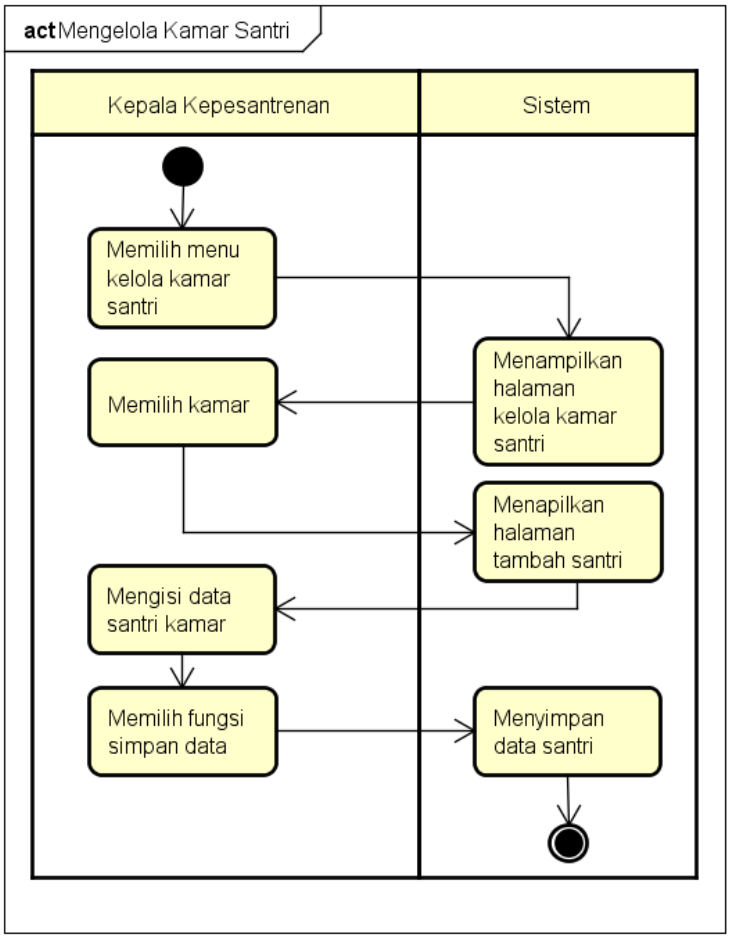
Gambar 4.9 menunjukkan diagram aktivitas pada proses mengelola kamar. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola kamar pada Tabel 4.8. Aktivitas dimulai ketika kepala kepesantrenan memilih menu kelola kamar. Kemudian sistem menampilkan halaman kelola kamar. Selanjutnya Kepala kepesantrenan memilih fungsi tambah kamar. Sistem menampilkan halaman tambah kamar dan kepala kepesantrenan mengisi data kamar. Kemudian kepala kepesantrenan memilih fungsi simpan data dan sistem menyimpan data kamar.



Gambar 4.9 *Activity Diagram* Mengelola Kamar

4.3.1.3 *Activity Diagram* Mengelola Kamar Santri

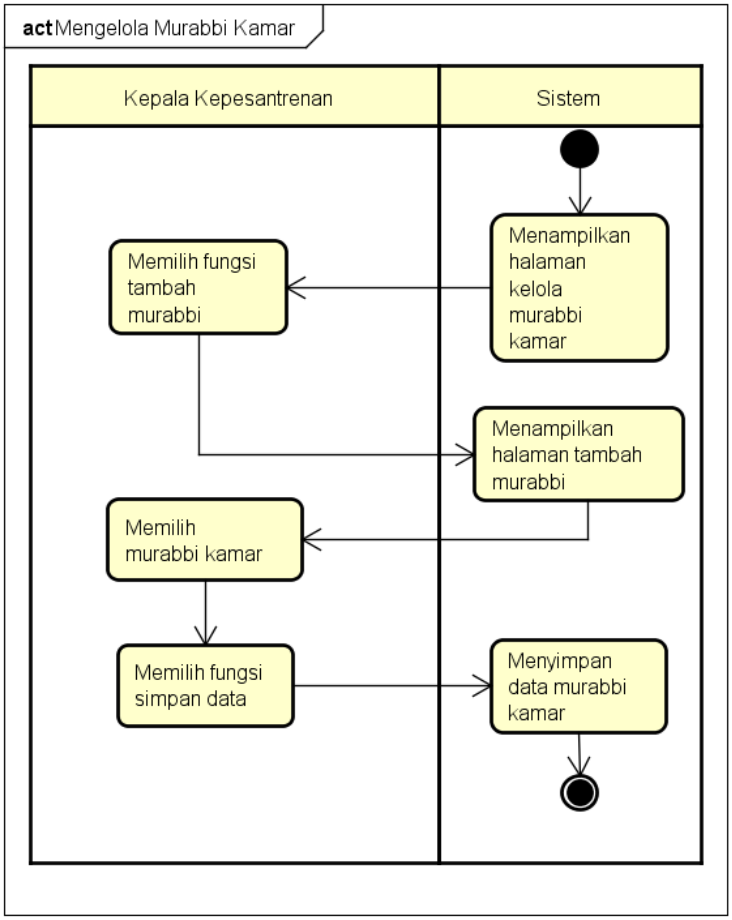
Gambar 4.10 menunjukkan diagram aktivitas pada proses mengelola kamar santri. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola kamar santri pada Tabel 4.9. Aktivitas dimulai dari sistem menampilkan halaman kelola kamar santri. Kepala kepesantrenan memilih kamar yang akan diisi santrinya. Kemudian sistem menampilkan halaman tambah santri. Kepala kepesantrenan memasukkan santri kamar tersebut kemudian memilih fungsi simpan data santri dan sistem menyimpan data.



Gambar 4.10 *Activity Diagram* Mengelola Kamar Santri

4.3.1.4 *Activity Diagram* Mengelola Murabbi Kamar

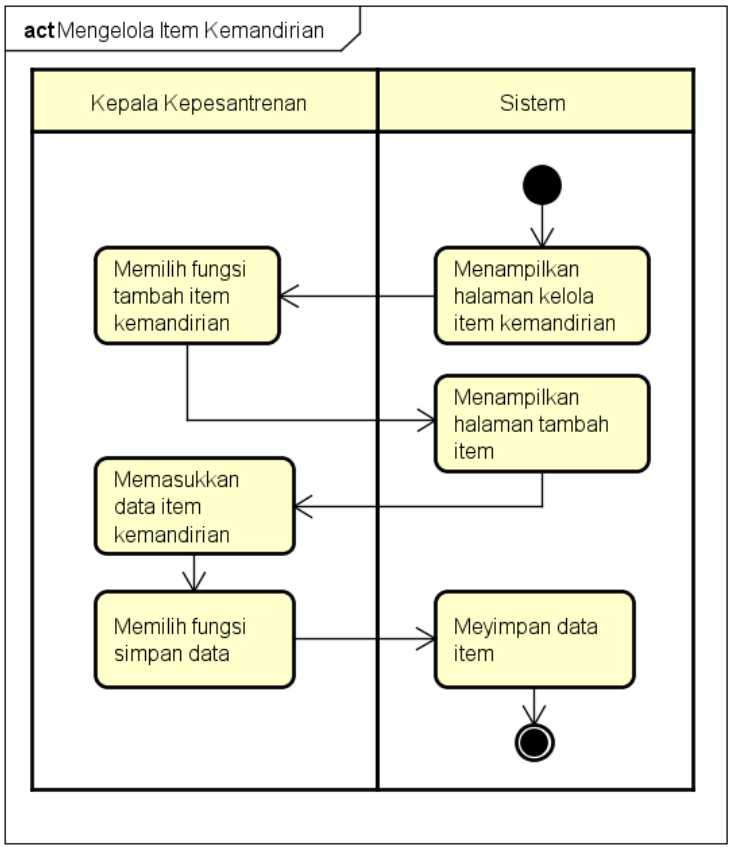
Gambar 4.11 menunjukkan diagram aktivitas pada proses mengelola murabbi kamar. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola murabbi kamar pada Tabel 4.9. Aktivitas dimulai dari sistem menampilkan halaman kelola murabbi kamar. Kepala kepesantrenan memilih fungsi tambah murabbi kamar. Kemudian sistem menampilkan halaman tambah murabbi. Kepala kepesantrenan memasukkan data murabbi beserta kamarnya kemudian memilih fungsi simpan data santri dan sistem menyimpan data.



Gambar 4.11 *Activity Diagram* Mengelola Murabbi Kamar

4.3.1.5 *Activity Diagram* Mengelola Item Kemandirian

Gambar 4.12 menunjukkan diagram aktivitas pada proses mengelola item kemandirian. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola item kemandirian pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola item kemandirian. Selanjutnya kepala kepesantrenan memilih fungsi tambah item kemandirian. Sistem menampilkan halaman tambah item kemudian kepala kepesantrenan memasukkan data item kemandirian. Kepala kepesantrenan memilih fungsi simpan data kemudian sistem menyimpan data yang sudah dimasukkan.

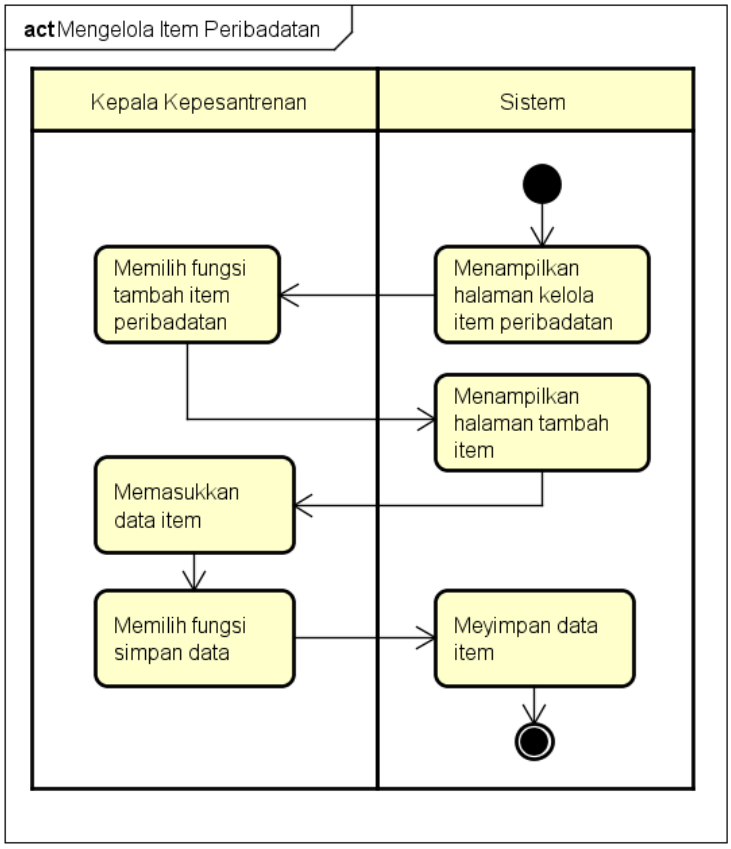


Gambar 4.12 *Activity Diagram* Mengelola Item Kemandirian

4.3.1.6 *Activity Diagram* Mengelola Item Peribadatan

4.3.1.7 *Activity Diagram* Mengelola Item Peribadatan

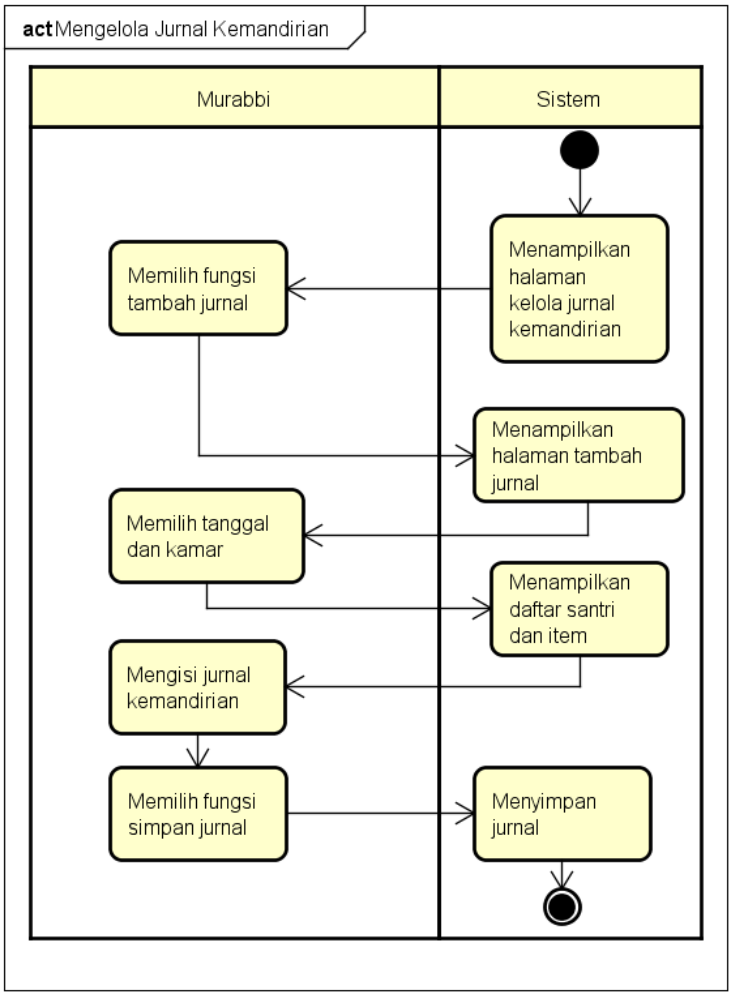
Gambar 4.13 menunjukkan diagram aktivitas pada proses mengelola item peribadatan. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola item peribadatan pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola item peribadatan. Selanjutnya kepala kepesantrenan memilih fungsi tambah item peribadatan. Sistem menampilkan halaman tambah item kemudian kepala kepesantrenan memasukkan data item peribadatan. Kepala kepesantrenan memilih fungsi simpan data kemudian sistem menyimpan data yang sudah dimasukkan.



Gambar 4.13 *Activity Diagram* Mengelola Item Peribadatan

4.3.1.8 *Activity Diagram* Mengelola Jurnal Kemandirian

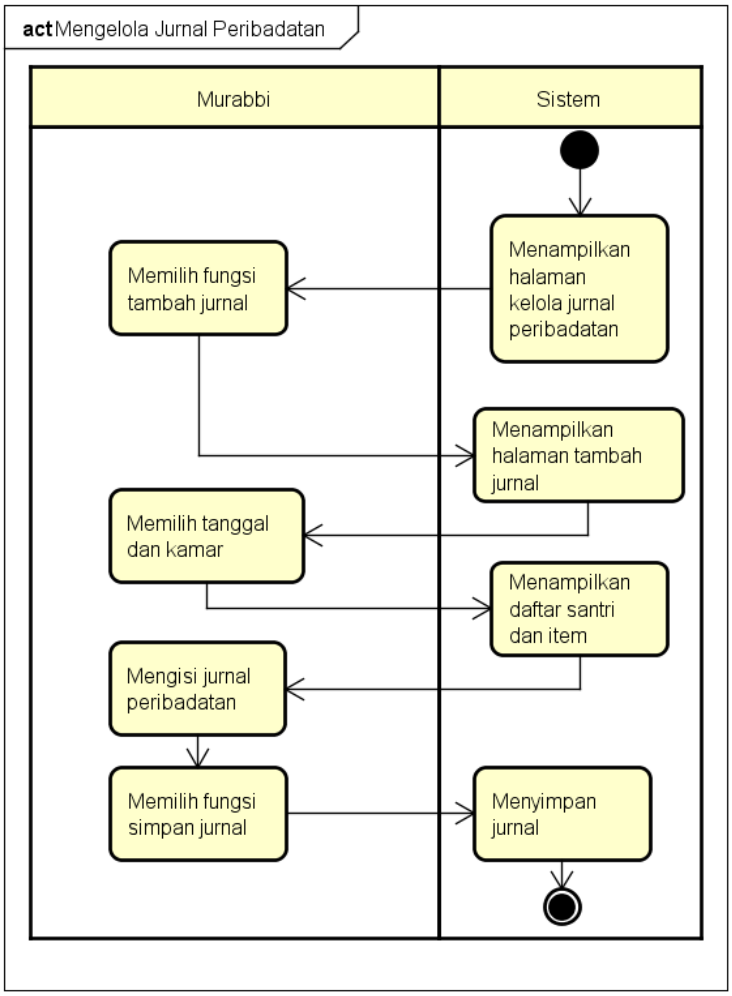
Gambar 4.14 menunjukkan diagram aktivitas pada proses mengelola jurnal kemandirian. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola jurnal kemandirian pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola jurnal kemandirian. Selanjutnya murabbi memilih fungsi tambah jurnal. Sistem menampilkan halaman tambah jurnal kemudian murabbi memilih tanggal dan kamar yang akan diisi. Sistem menampilkan daftar santri kamar yang dipilih. Kemudian murabbi memasukkan data jurnal dan memilih fungsi simpan data. Selanjutnya sistem menyimpan data yang sudah dimasukkan.



Gambar 4.14 *Activity Diagram* Mengelola Jurnal Kemandirian

4.3.1.9 *Activity Diagram* Mengelola Jurnal Peribadatan

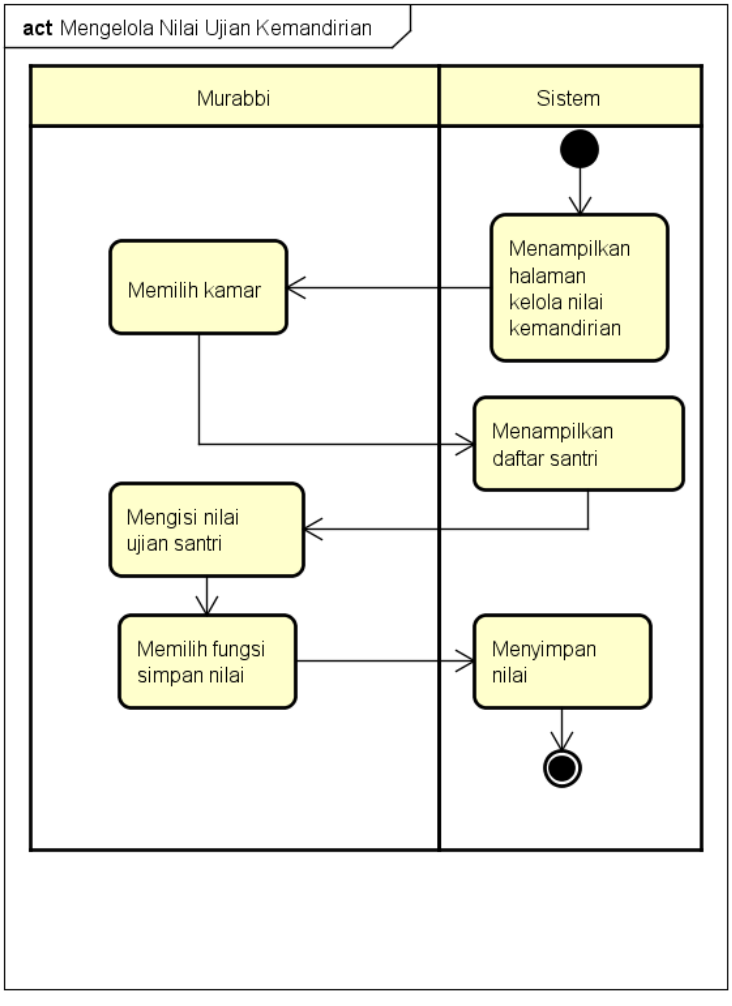
Gambar 4.15 menunjukkan diagram aktivitas pada proses mengelola jurnal peribadatan. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola jurnal peribadatan pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola jurnal peribadatan. Selanjutnya murabbi memilih fungsi tambah jurnal. Sistem menampilkan halaman tambah jurnal kemudian murabbi memilih tanggal dan kamar yang akan diisi. Sistem menampilkan daftar santri kamar yang dipilih. Kemudian murabbi memasukkan data jurnal dan memilih fungsi simpan data. Selanjutnya sistem menyimpan data yang sudah dimasukkan.



Gambar 4.15 *Activity Diagram* Mengelola Jurnal Peribadatan

4.3.1.10 *Activity Diagram* Mengelola Nilai Ujian Kemandirian

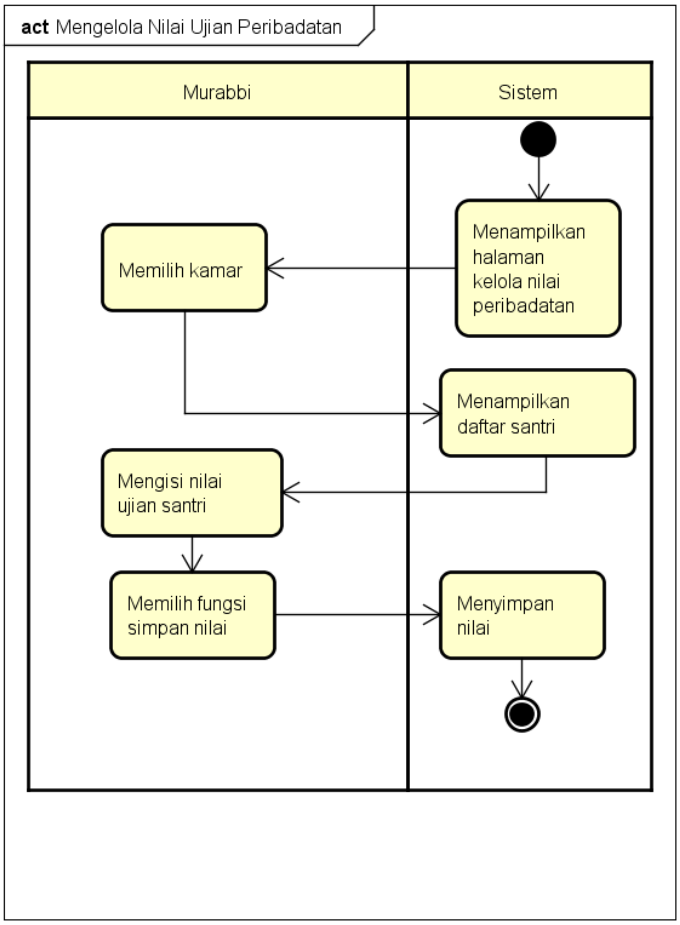
Gambar 4.16 menunjukkan diagram aktivitas pada proses mengelola nilai ujian kemandirian. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola nilai ujian kemandirian pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola nilai ujian kemandirian. Selanjutnya murabbi memilih kamar yang akan diisi nilainya. Sistem menampilkan halaman tambah nilai pada kamar yang dipilih. Kemudian murabbi memasukkan nilai setiap anak dan memilih fungsi simpan data. Selanjutnya sistem menyimpan data yang sudah dimasukkan.



Gambar 4.16 *Activity Diagram* Mengelola Nilai Ujian Kemandirian

4.3.1.12 *Activity Diagram* Mengelola Nilai Ujian Peribadatan

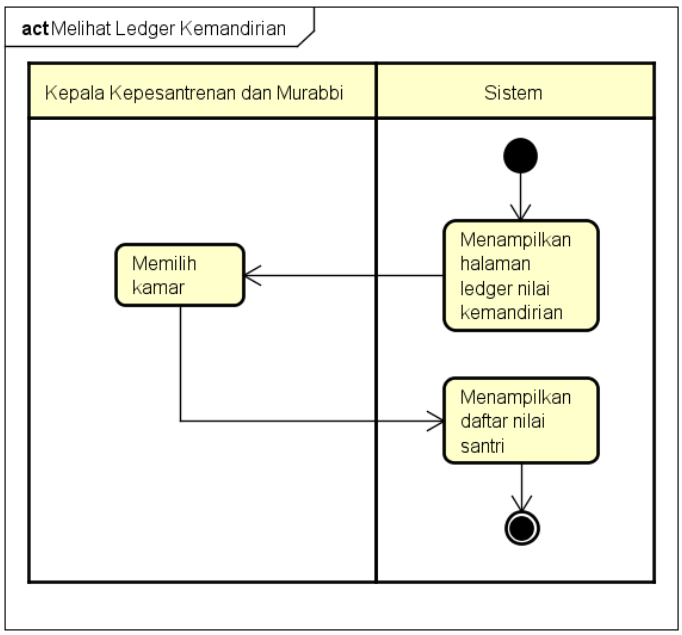
Gambar 4.17 menunjukkan diagram aktivitas pada proses mengelola nilai ujian peribadatan. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* mengelola nilai ujian peribadatan pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman kelola nilai ujian peribadatan. Selanjutnya murabbi memilih kamar yang akan diisi nilainya. Sistem menampilkan halaman tambah nilai pada kamar yang dipilih. Kemudian murabbi memasukkan nilai setiap anak dan memilih fungsi simpan data. Selanjutnya sistem menyimpan data yang sudah dimasukkan.



Gambar 4.17 *Activity Diagram* Mengelola Nilai Ujian Peribadatan

4.3.1.13 *Activity Diagram* Melihat Ledger Kemandirian

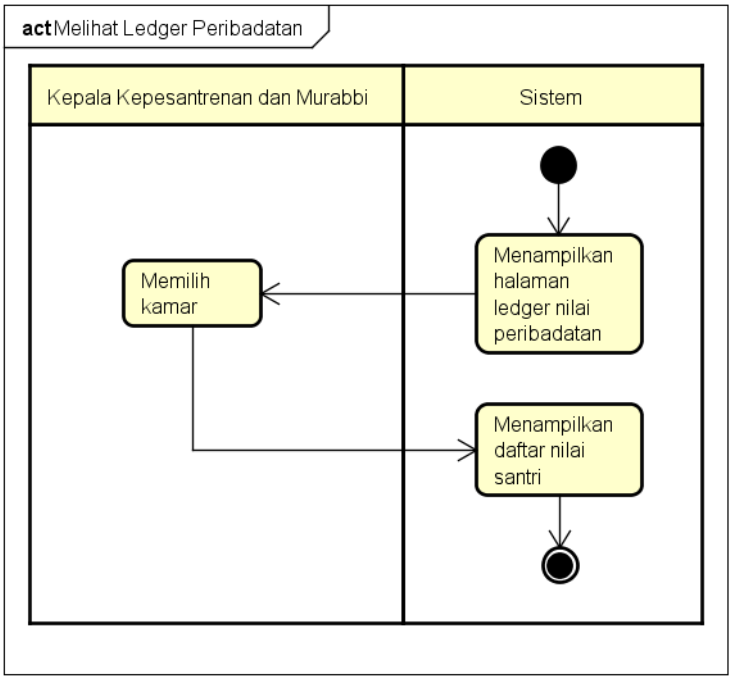
Gambar 4.18 menunjukkan diagram aktivitas pada proses melihat ledger kemandirian. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* melihat ledger kemandirian pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman ledger kemandirian. Selanjutnya murabbi atau kepala kepesantrenan memilih kamar yang akan dilihat ledgernya. Sistem menampilkan daftar nilai kemandirian setiap santri pada kamar yang dipilih.



Gambar 4.18 *Activity Diagram* Melihat Ledger Kemandirian

4.3.1.14 *Activity Diagram* Melihat Ledger Peribadatan

Gambar 4.19 menunjukkan diagram aktivitas pada proses melihat ledger peribadatan. *Activity diagram* tersebut menggambarkan bagaimana aktivitas yang terjadi sesuai *use case scenario* melihat ledger peribadatan pada **Error! Reference source not found.**. Aktivitas dimulai dari sistem menampilkan halaman ledger peribadatan. Selanjutnya murabbi atau kepala kepesantrenan memilih kamar yang akan dilihat ledgernya. Sistem menampilkan daftar nilai peribadatan setiap santri pada kamar yang dipilih.

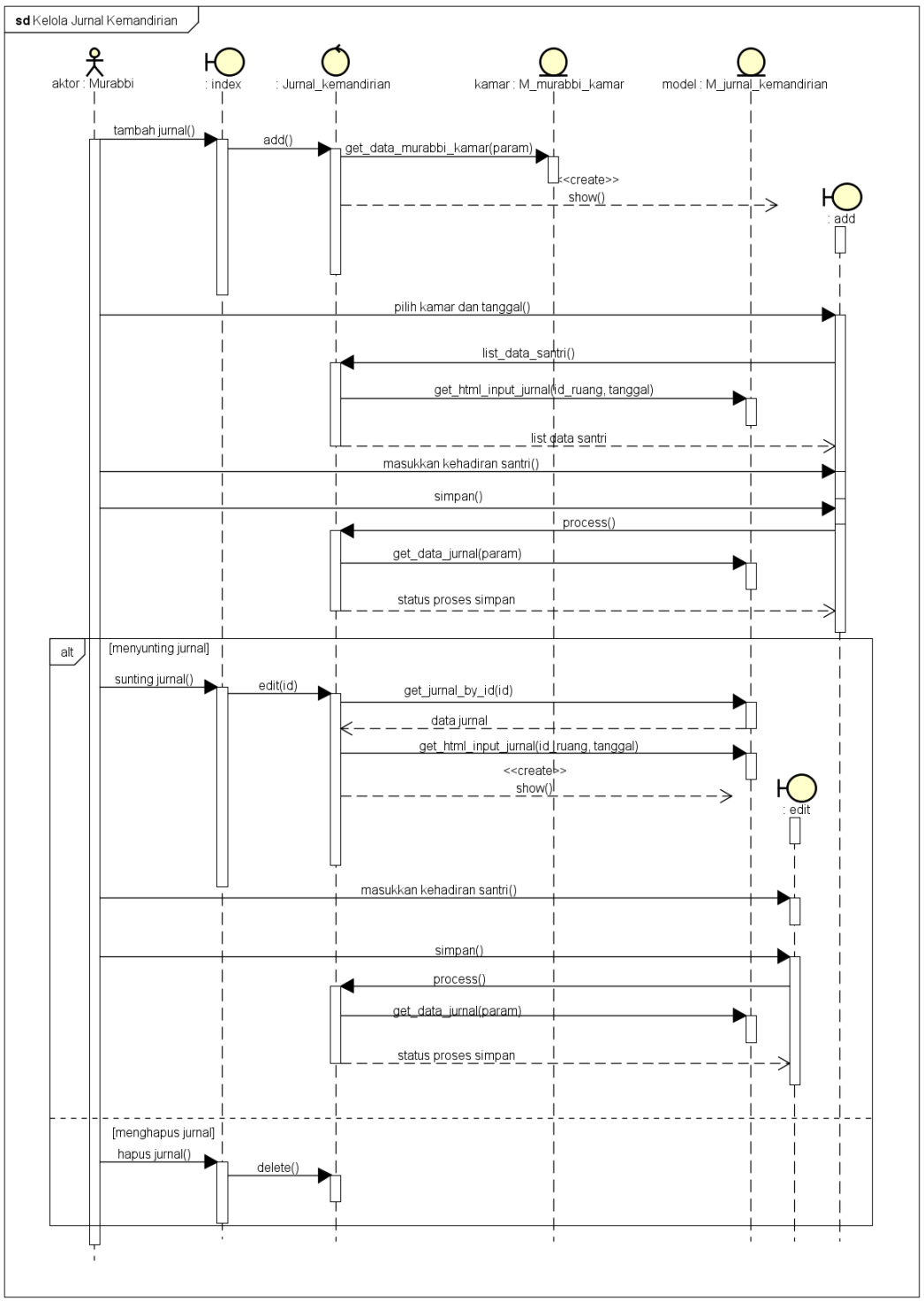


Gambar 4.19 *Activity Diagram* Melihat Ledger Peribadatan

4.3.2 Sequence Diagram

Sequence diagram menjelaskan interaksi objek dan menunjukkan komunikasi di antara objek-objek tersebut. *Sequence digaram* digunakan untuk menggambarkan perilaku skenario dan bagaimana entitas berinteraksi dengan sistem, termasuk pesan yang digunakan selama interaksi. Semua pesan dijelaskan dalam urutan eksekusi. *Sequence digaram* terkait erat dengan *Use case* digaram, dimana satu *use case* menjadi satu *sequence diagram*. Pada bagian ini akan dijelaskan *sequence diagram* pada Sistem Informasi Manajemen Pengasuhan Santri di Pesantren. *Sequence diagram* yang digambarkan hanya *use case* yang menjadi solusi utama dari permasalahan.

4.3.2.1 *Sequence Diagram* Mengelola Jurnal Kemandirian



Gambar 4.20 *Sequence Diagram* Mengelola Jurnal Kemandirian

Gambar 4.20 menunjukkan *sequence diagram* mengelola jurnal kemandirian. Urutan dimulai ketika aktor murabbi menjalankan perintah tambah jurnal pada halaman *index*. Halaman *indeks* akan memanggil fungsi *add()* pada *class* Jurnal_kemandirian untuk menampilkan halaman tambah jurnal. Pada fungsi

add() menjalankan fungsi get_data_murabbi_kamar() pada objek kamar untuk mendapatkan data kamar yang diasuh murabbi tersebut. Setelah data kamar didapatkan, fungsi add() akan menampilkan halaman tambah jurnal bernama *add.* Pengguna diminta untuk memilih kamar dan tanggal jurnal yang akan dimasukkan. Setelah kamar dan tanggal dipilih, halaman *add* memanggil fungsi list_data_santri() pada *class* Jurnal_kemandirian untuk mendapatkan kolom masukkan kehadiran santri. Fungsi list_data_santri() melanjutkan memanggil fungsi get_html_input_jurnal (id_kamar, tanggal) dari objek bernama model untuk mendapatkan data santri dan membuat kolom masukkan. Kemudian kolom masukan ditampilkan untuk diisi pengguna dan disimpan lalu diproses pada *class* Jurnal_kemandirian.

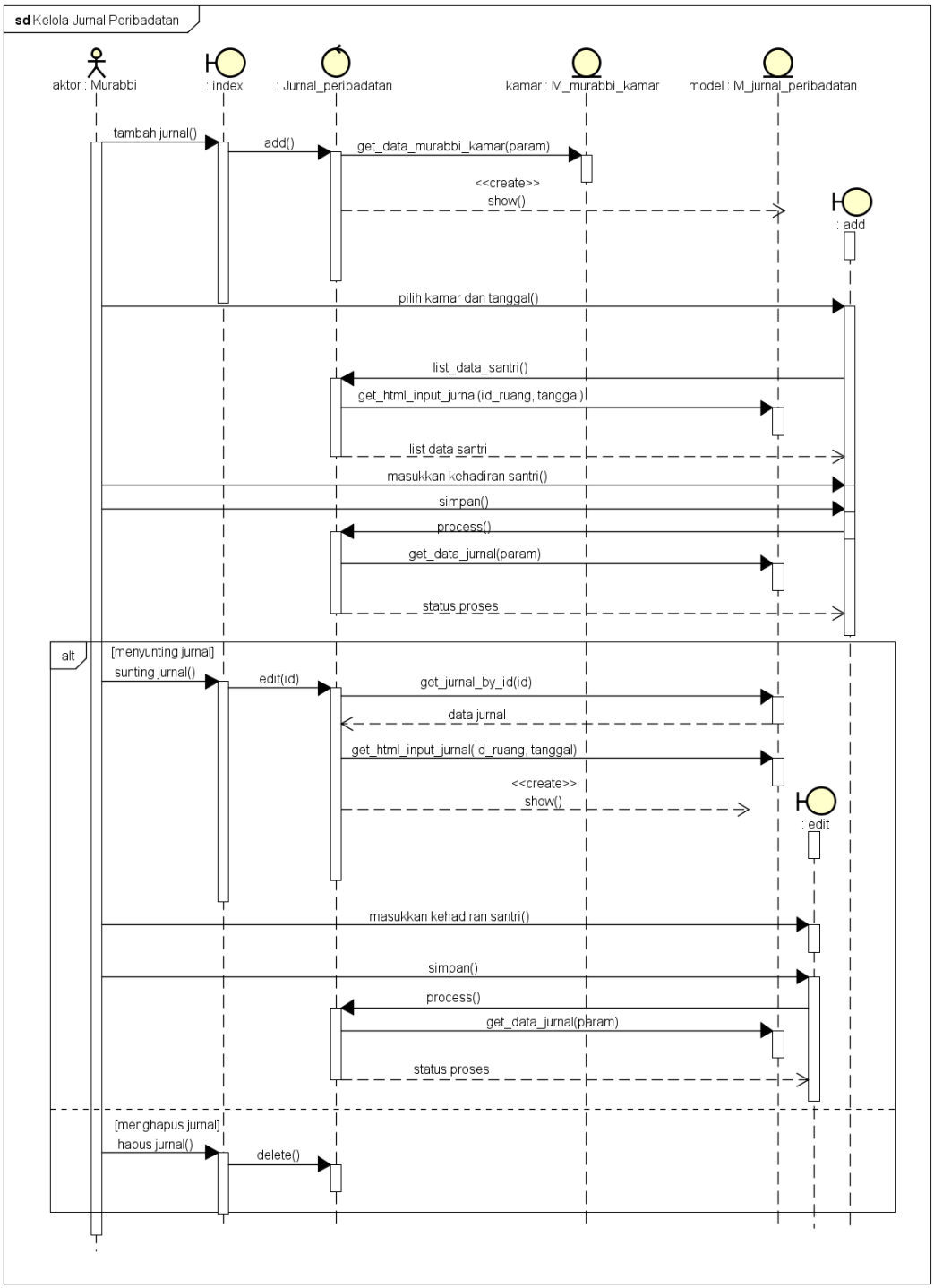
Selain urutan yang sudah dijelaskan sebelumnya, terdapat urutan alternatif ketika pengguna akan mengubah data kehadiran dan menghapus data jurnal. Urutan tersebut digambarkan pada alternatif menyunting dan menghapus jurnal. Menyunting jurnal dimulai ketika aktor memilih data jurnal untuk disunting pada halaman *index*, kemudian halaman *index* akan memanggil fungsi *edit(id)* pada class Jurnal_kemandirian untuk menampilkan halaman edit. Setelah halaman edit ditampilkan, pengguna dapat mengubah data kehadiran santri lalu menyimpannya. Urutan alternatif menghapus jurnal dimulai ketika pengguna memilih data jurnal yang akan dihapus pada halaman *index* kemudian akan memanggil fungsi *delete()* pada class Jurnal_kemandirian untuk menghapus data jurnal yang dipilih.

4.3.2.2 *Sequence Diagram* Mengelola Jurnal Peribadatan

Gambar 4.21 adalah gambar *sequence diagram* mengelola jurnal peribadatan. Urutan dimulai ketika aktor murabbi menjalankan perintah tambah jurnal pada halaman *index*. Halaman *indeks* akan memanggil fungsi *add()* pada *class* Jurnal_peribadatan untuk menampilkan halaman tambah jurnal. Pada fungsi *add()* menjalankan fungsi get_data_murabbi_kamar() pada objek kamar untuk mendapatkan data kamar yang diasuh murabbi tersebut. Setelah data kamar didapatkan, fungsi add() akan menampilkan halaman tambah jurnal bernama *add.* Pengguna diminta untuk memilih kamar dan tanggal jurnal yang akan dimasukkan. Setelah kamar dan tanggal dipilih, halaman *add* memanggil fungsi list_data_santri() pada *class* Jurnal_peribadatan untuk mendapatkan kolom masukkan kehadiran santri. Fungsi list_data_santri() melanjutkan memanggil fungsi get_html_input_jurnal (id_kamar, tanggal) dari objek bernama model untuk mendapatkan data santri dan membuat kolom masukkan. Kemudian kolom masukan ditampilkan untuk diisi pengguna dan disimpan lalu diproses pada *class* Jurnal_peribadatan.

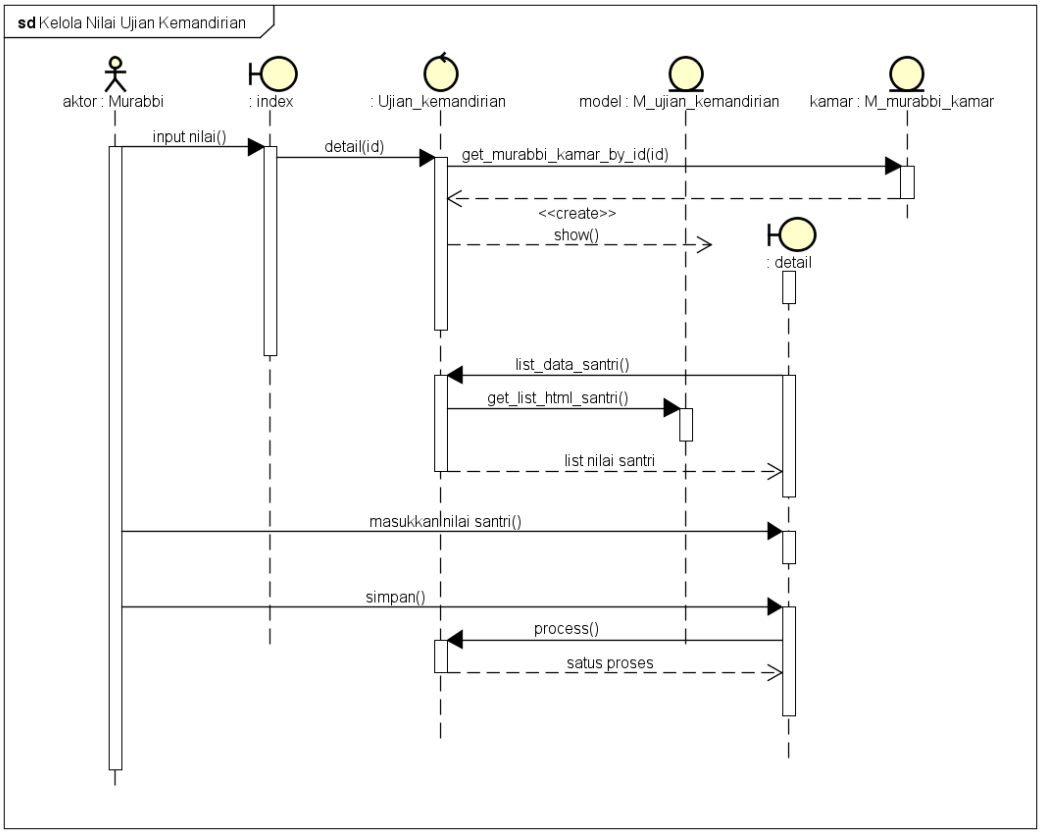
Selain urutan yang sudah dijelaskan sebelumnya, terdapat urutan alternatif ketika pengguna akan mengubah data kehadiran dan menghapus data jurnal. Urutan tersebut digambarkan pada alternatif menyunting dan menghapus jurnal. Menyunting jurnal dimulai ketika aktor memilih data jurnal untuk disunting pada halaman *index*, kemudian halaman *index* akan memanggil fungsi *edit(id)* pada class Jurnal_kemandirian untuk menampilkan halaman edit. Setelah halaman edit

ditampilkan, pengguna dapat mengubah data kehadiran santri lalu menyimpannya. Urutan alternatif menghapus jurnal dimulai ketika pengguna memilih data jurnal yang akan dihapus pada halaman *index* kemudian akan memanggil fungsi *delete()* pada class Jurnal_kemandirian untuk menghapus data jurnal yang dipilih.



Gambar 4.21 *Sequence Diagram* Mengelola Jurnal Peribadatan

4.3.2.3 *Sequence Diagram* Mengelola Nilai Ujian Kemandirian

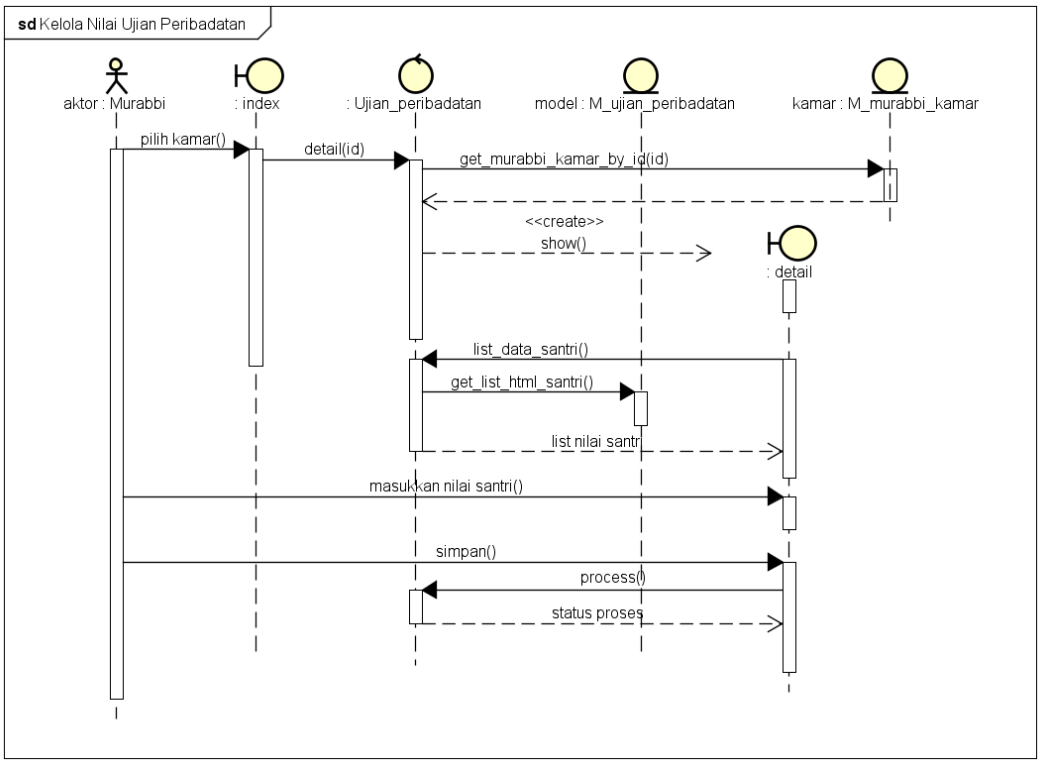


Gambar 4.22 *Sequence Diagram* Mengelola Nilai Ujian Kemandirian

Urutan aktivitas mengelola nilai ujian kemandirian ditunjukkan pada Gambar 4.22. Urutan dimulai saat aktor menjalankan menu input nilai. Kemudian *controller* akan menampilkan halaman yang berisi informasi kamar yang dipilih dan daftar santri untuk dinilai. Setelah itu, aktor memasukkan nilai tiap santri dan menyimpannya. *Controller* kemudian memproses data yang dikirim untuk disimpan di basis data.

4.3.2.4 *Sequence Diagram* Mengelola Nilai Ujian Peribadatan

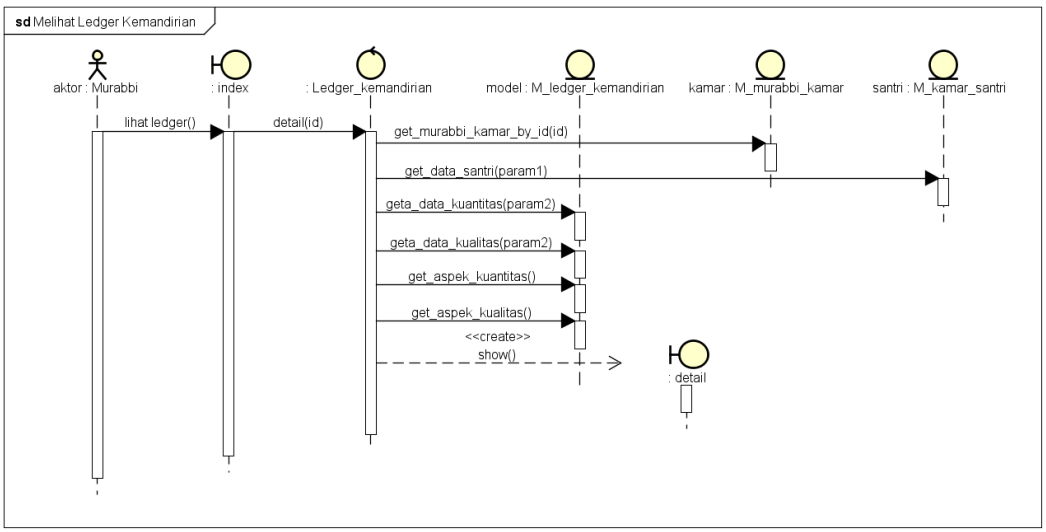
Urutan aktivitas mengelola nilai ujian peribadatan ditunjukkan pada Gambar 4.33. Urutan dimulai saat aktor menjalankan menu *input* nilai. Kemudian *controller* akan menampilkan halaman yang berisi informasi kamar yang dipilih dan daftar santri untuk dinilai. Setelah itu, aktor memasukkan nilai tiap santri dan menyimpannya. *Controller* kemudian memproses data yang dikirim untuk disimpan di basis data.



Gambar 4.23 *Sequence Diagram* Mengelola Nilai Ujian Peribadatan

4.3.2.5 *Sequence Diagram* Melihat Ledger Kemandirian

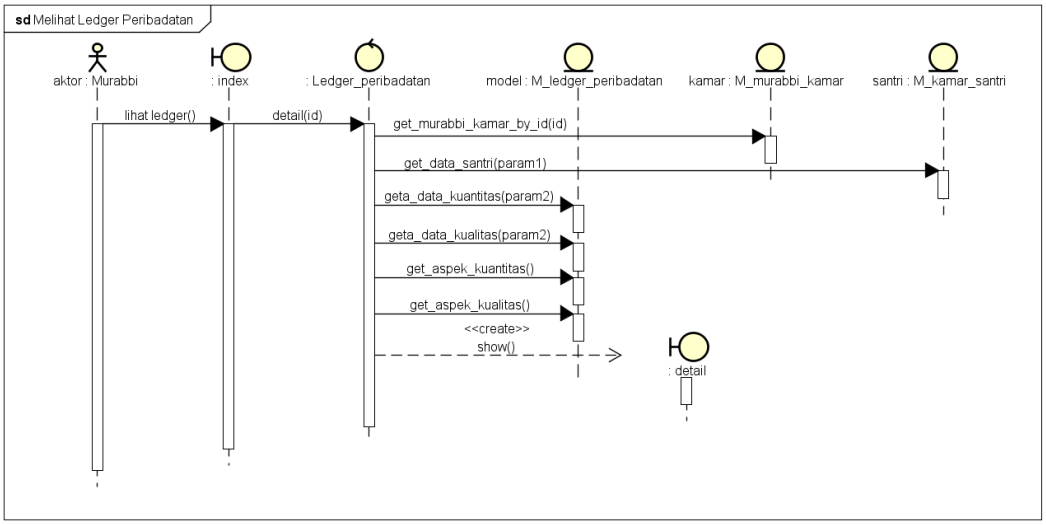
Sequence diagram melihat ledger kemandirian ditunjukkan pada Gambar 4.34. Urutan dimulai saat aktor menjalankan menu lihat ledger. Kemudian *controller* akan melakukan *request* ke beberapa *model* untuk mendapatkan data kamar, santri dan nilai kemandirian. Setelah mendapatkan data yang diminta, data tersebut ditampilkan pada halaman detail ledger.



Gambar 4.24 *Sequence Diagram* Melihat Ledger Kemandirian

4.3.2.6 *Sequence Diagram* Melihat Ledger Peribadatan

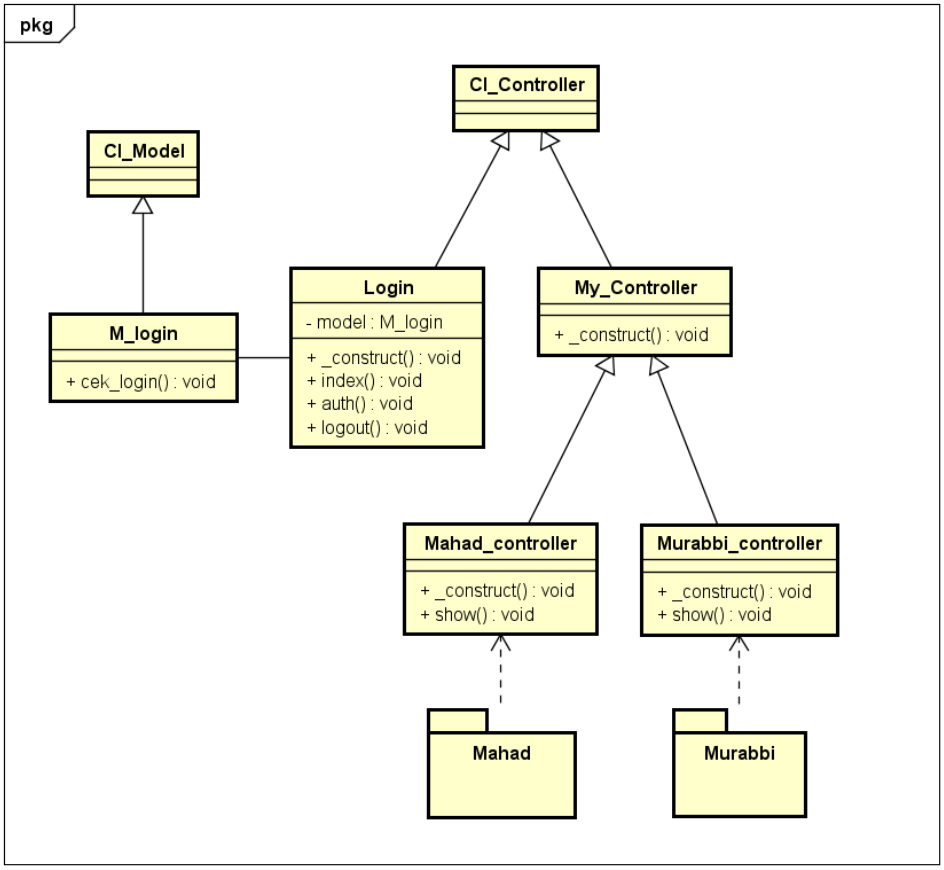
Sequence diagram melihat ledger peribadatan ditunjukkan pada Gambar 4.34. Urutan dimulai saat aktor menjalankan menu lihat ledger. Kemudian *controller* akan melakukan *request* ke beberapa *model* untuk mendapatkan data kamar, santri dan nilai peribadatan. Setelah mendapatkan data yang diminta, data tersebut ditampilkan pada halaman detail ledger.



Gambar 4.25 *Sequence Diagram* Melihat Ledger Peribadatan

4.3.3 *Class Diagram*

Pengembangan sistem informasi manajemen pengasuhan santri menggunakan pendekatan arsitektur MVC, sehingga perancangan *class digaram* dibagi menjadi tiga bagian, yaitu *model, view* dan *controller. Model* bertanggung jawab atas persiapan, pemrosesan dan pengorganisasian data (dari *database*) sesuai dengan instruksi *controller*. View bertugas menampilkan informasi kepada pengguna sesuai instruksi dari *controller*. *Controller* bertugas mengatur apa yang harus dilakukan *model*, dan *view* mana yang harus ditampilkan. Gambar 4.27 merupakan *class diagram* sistem informasi manajemen pengasuhan santri.

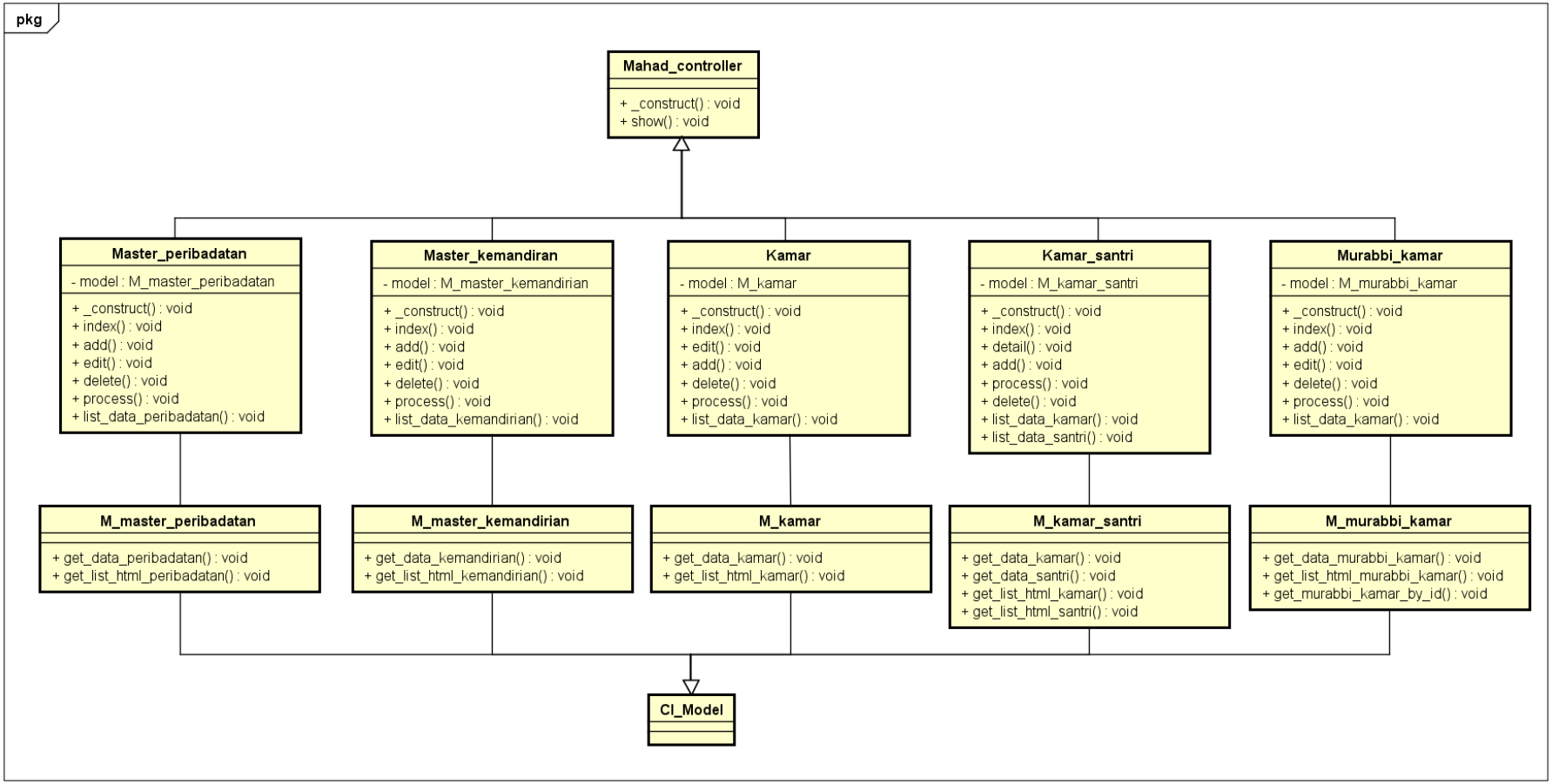


Gambar 4.26 Class Diagram Sistem Informasi Manajemen Kepengasuhan

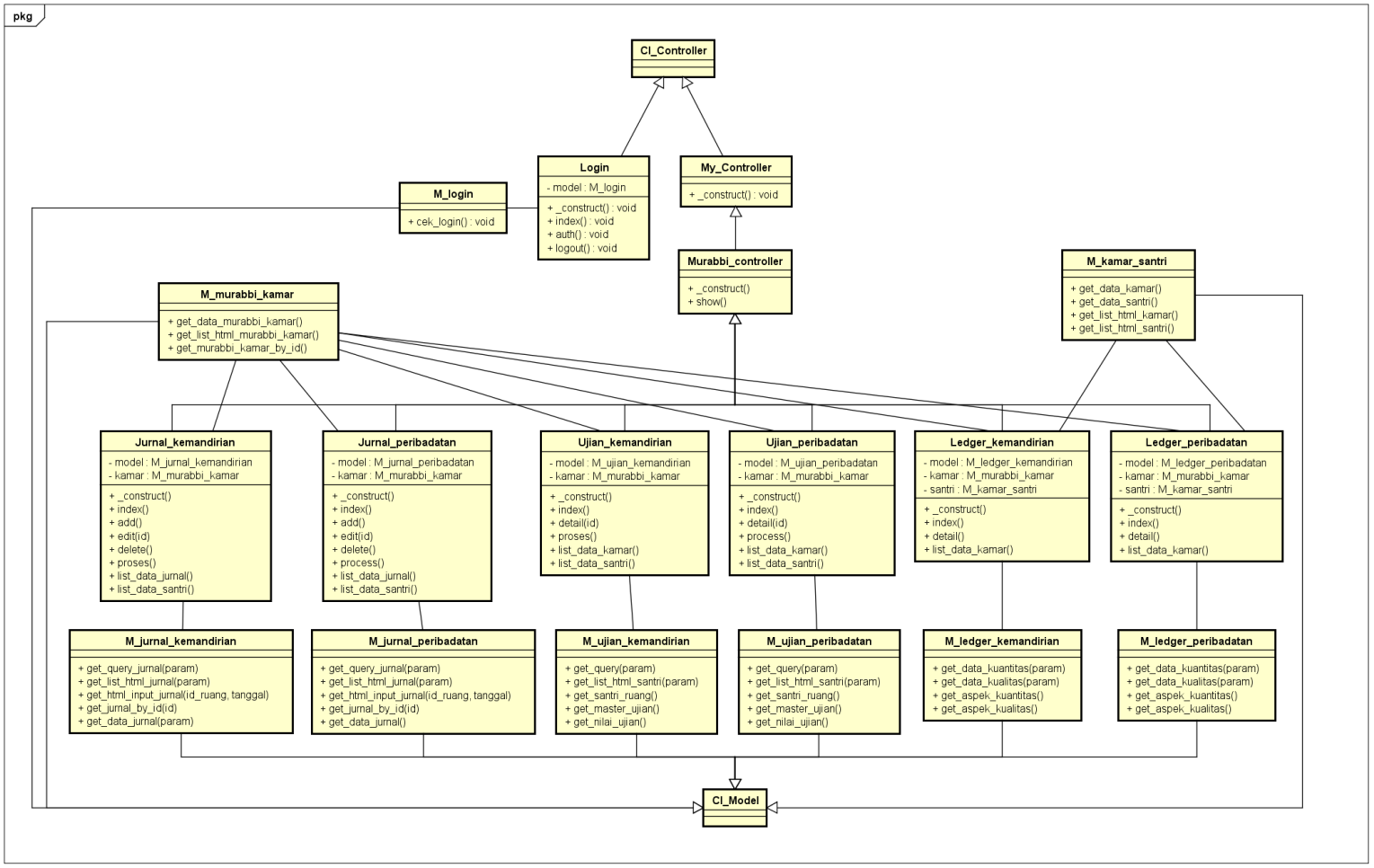
Gambar 4.26 merupakan *class digaram* diagram sistem informasi manajemen kepengasuhan. Sistem dibagi menjadi dua modul yaitu mahad dan murabbi. Modul mahad ditujukan untuk hak akses kepala kepesantrenan. Sedangkan murabbi digunakan untuk hak akses pengasuh.

4.3.3.1 *Class digram* modul mahad

Gambar 4.27 merupakan *class diagram* untuk modul mahad. Terdapat *class* Master_peribadatan, Master_kemandirian, Kamar, Kamar_santri dan Murabbi_kamar yang merupakan *controller*. Controller pada modul mahad memiliki hubungan generalisasi dengan *class* Mahad_controller agar mewarisi atribut dan *method* dari *class* tersebut. Selain itu terdapat class M_master_peribadatan, M_master_kemandirian, M_kamar, M_kamar_santri dan M_kamar_murabbi sebagai model.



Gambar 4.27 *Class Diagram* Modul Mahad



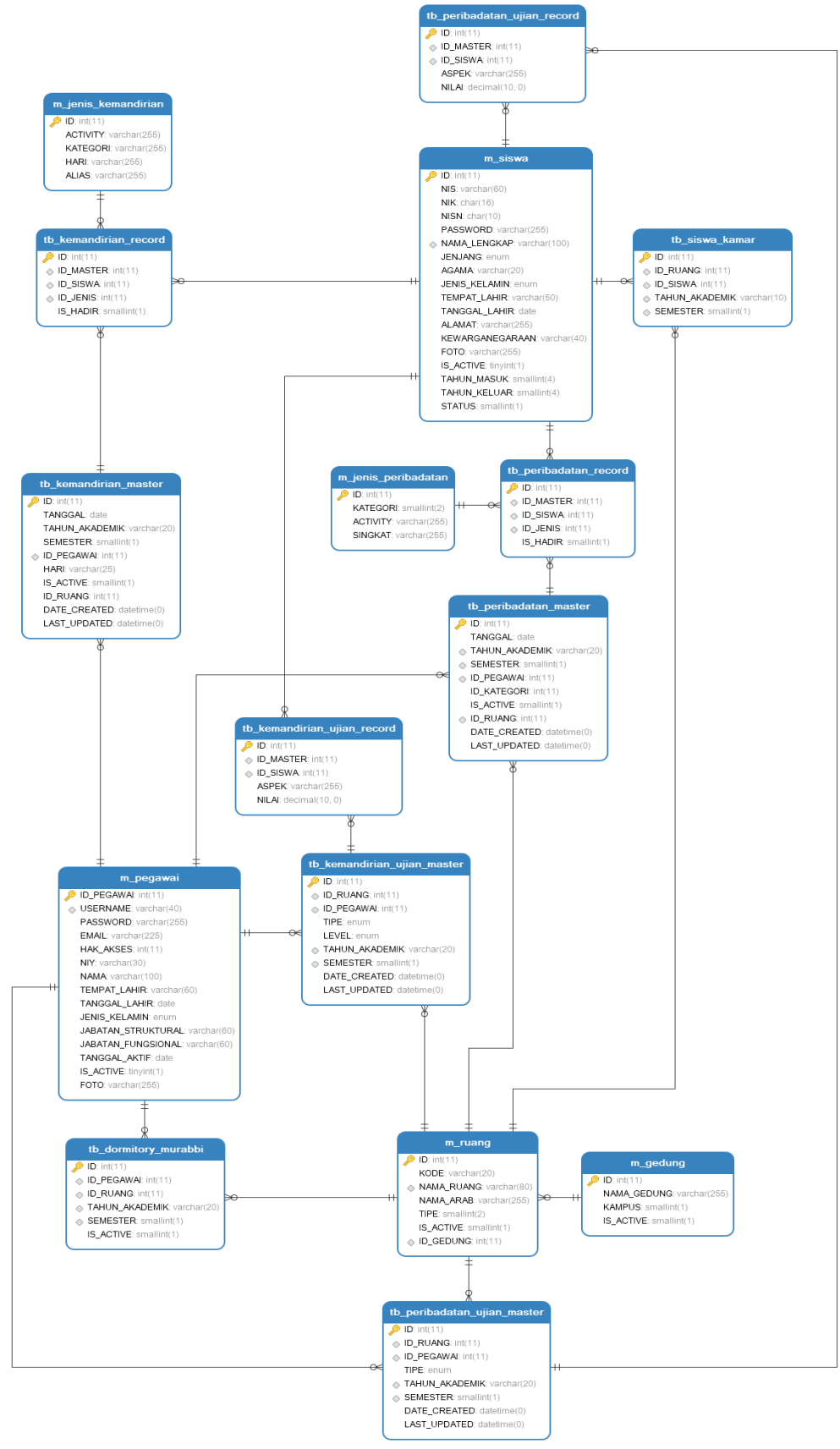
Gambar 4.28 *Class Diagram* Modul Murabbi

4.3.3.1 *Class digram* modul murabbi

Gambar 4.28 merupakan *class diagram* untuk modul murabbi. Pada modul murabbi memiliki beberapa *class*. Jurnal_peribadatan, Jurnal_kemandirian, Ujian_peribadatan, Ujian_kemandirian, Ledger_peribadatan dan Ledger_kemandirian.

4.3.4 Pemetaan *Class Diagram* ke Model Relasional

Setelah melakukan analisis *class diagram*, dilakukan pemetaan *class diagram* ke model relasional. Model relasional basis data dimodelkan dengan *physical entitiy relationship diagram (ERD). Physical* ERD merepresentasikan desain aktual dari hubungan basis data. Pada perancangan ERD juga didefinisikan tipe data yang diperlukan untuk kolom entitas. Selain itu, *primary, foreign key*, dan *constraints* juga didefinisikan pada model. Gambar 4.29 merupakan model relasional *physical data model (PDM)* hasil dari pemetaan *class diagram*.



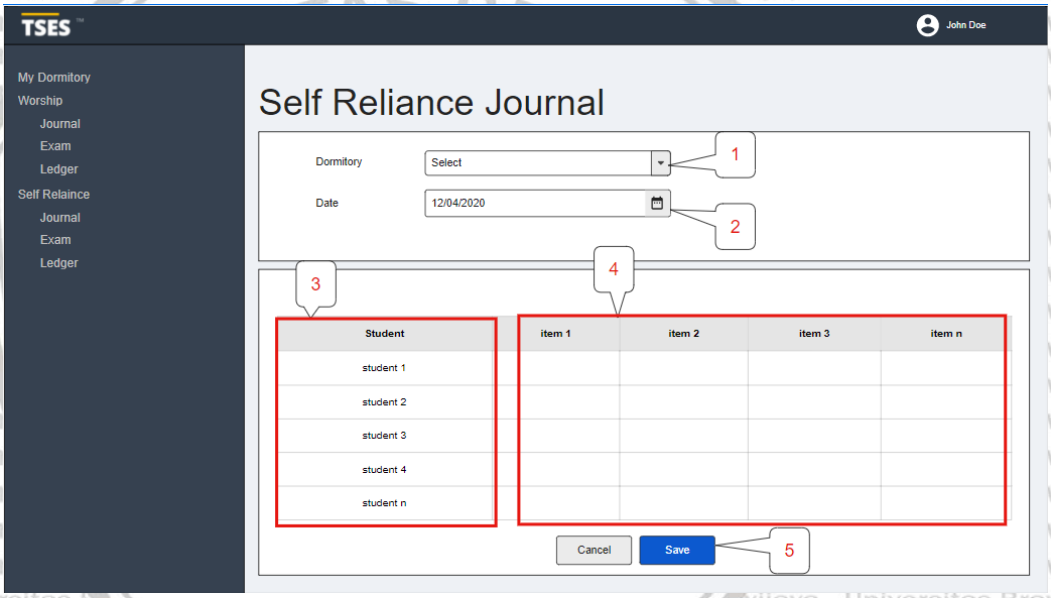
Gambar 4.29 *Physical Data Model*

4.3.5 Antarmuka Sistem

Antarmuka pengguna merupakan pembatas yang menghubungkan sistem dengan pengguna melalui perangkat yang digunakan. Antarmuka sistem informasi kepengasuhan secara umum dibagi menjadi dua modul, pertama modul ma'had yaitu antarmuka untuk hak akses kepala kepesantrenan. Modul ma'had menampilkan fitur-fitur yang digunakan oleh kepala kepesantrenan. Kedua, modul murabbi yaitu antarmuka sistem yang ditampilkan jika hak akses pengguna adalah murabbi.

4.3.5.1 Antarmuka mengelola jurnal kemandirian

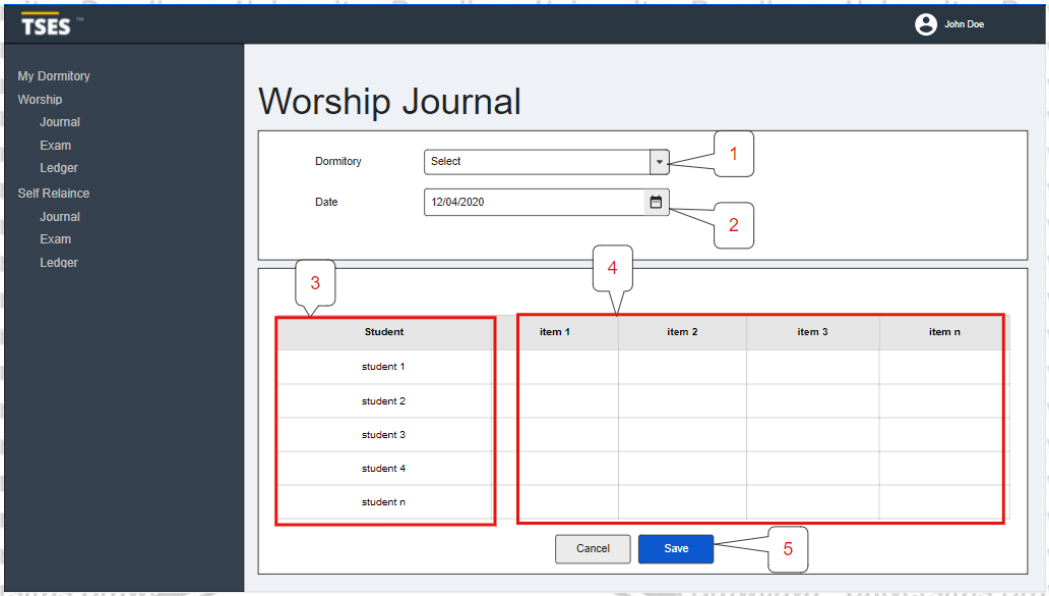
Antarmuka mengelola jurnal kemandirian adalah halaman yang ditampilkan sistem ketika pengguna akan melakukan pengolahan jurnal kemandirian. Gambar 4.30 menunjukkan antarmuka menambah jurnal kemandirian. Antarmuka menambah jurnal kemandirian memiliki beberapa komponen di antaranya, kolom masukkan pilih kamar (1), tanggal (2), daftar santri (3) dan kolom item (4) yang harus diisi serta tombol untuk menyimpan data jurnal (5).



Gambar 4.30 Rancangan antarmuka Menambah Jurnal Kemandirian

4.3.5.2 Antarmuka mengelola jurnal peribadatan

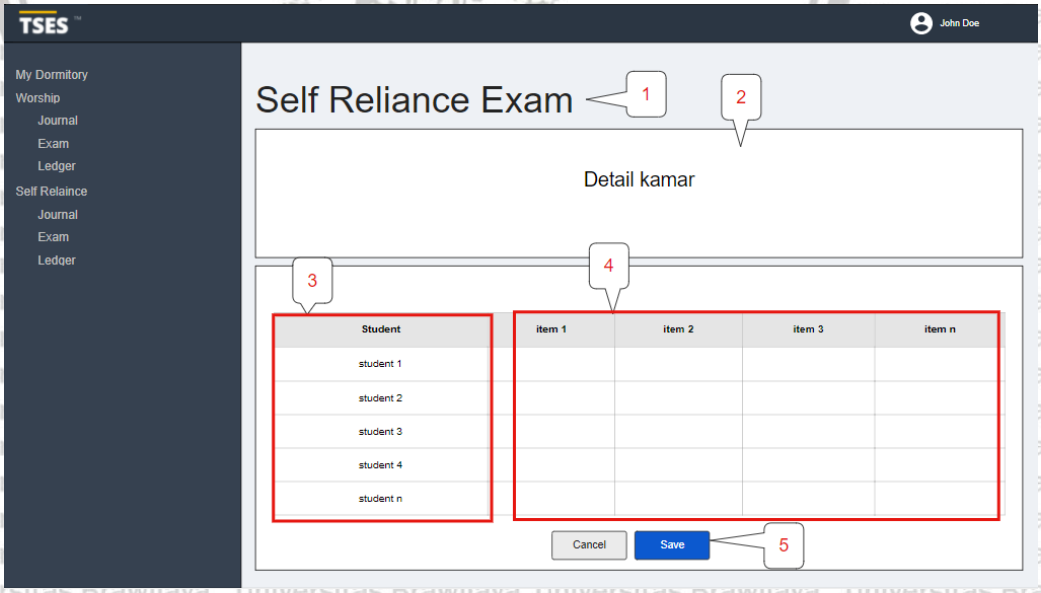
Antarmuka mengelola jurnal peribadatan adalah halaman yang ditampilkan sistem ketika pengguna akan melakukan pengolahan jurnal peribadatan. Gambar 4.31 menunjukkan antarmuka menambah jurnal peribadatan. Antarmuka menambah jurnal kemandirian memiliki beberapa komponen di antaranya, kolom masukkan pilih kamar (1), tanggal (2), daftar santri (3) dan kolom item (4) yang harus diisi serta tombol untuk menyimpan data jurnal (5).



Gambar 4.31 Rancangan antarmuka Menambah Jurnal Peribadatan

4.3.5.3 Antarmuka mengelola nilai ujian kemandirian

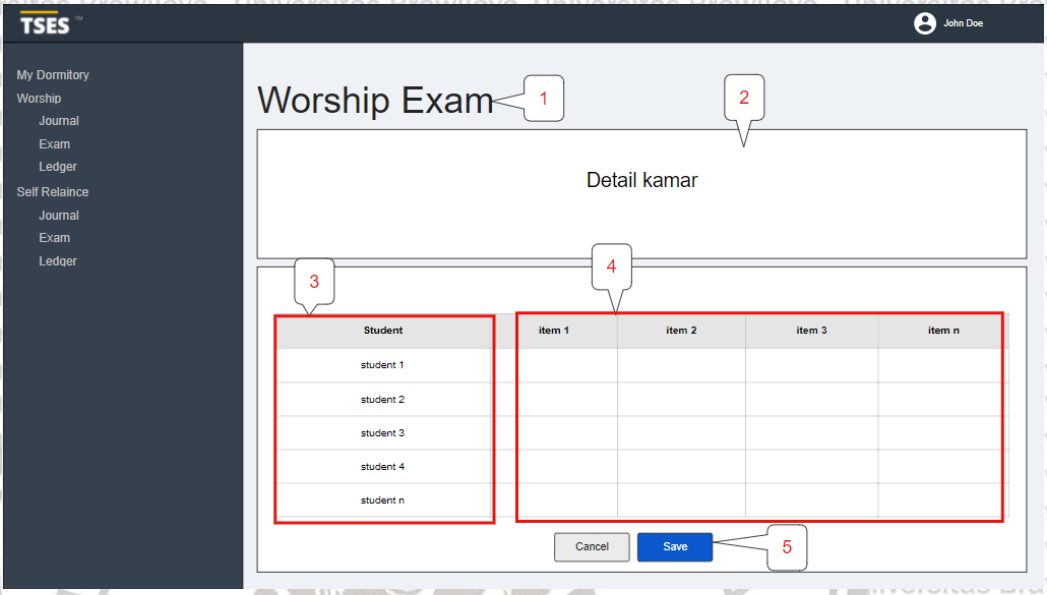
Antarmuka mengelola nilai ujian kemandirian adalah halaman yang ditampilkan sistem ketika pengguna akan melakukan pengolahan nilai peribadatan. Gambar 4.33 menunjukkan antarmuka memasukkan nilai ujian kemandirian. Antarmuka memasukkan nilai ujian kemandirian memiliki beberapa komponen di antaranya, judul halaman (1), informasi kamar yang diisi (2), daftar santri (3) dan kolom item (4) yang harus diisi serta tombol untuk menyimpan data nilai (5).



Gambar 4.32 Rancangan antarmuka Memasukkan Nilai Ujian Kemandirian

4.3.5.4 Antarmuka mengelola nilai ujian peribadatan

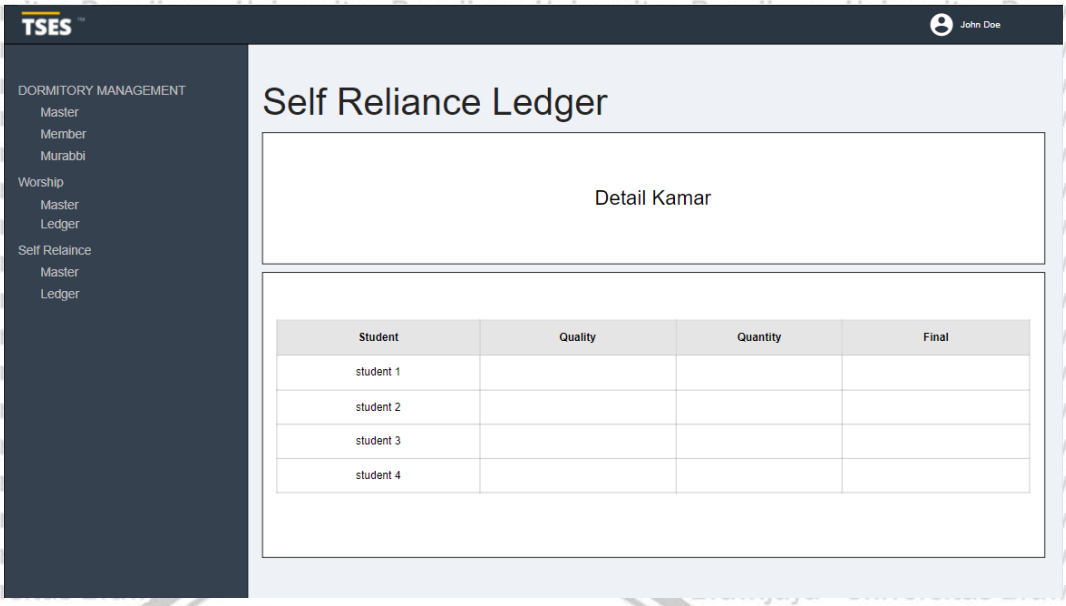
Antarmuka mengelola ujian peribadatan adalah halaman yang ditampilkan sistem ketika pengguna akan melakukan pengolahan nilai ujian peribadatan. Gambar 4.33 menunjukkan antarmuka memasukkan nilai ujian peribadatan. Antarmuka memasukkan nilai peribadatan memiliki beberapa komponen di antaranya, judul halaman (1), informasi kamar yang diisi (2), daftar santri (3) dan kolom item (4) yang harus diisi serta tombol untuk menyimpan data nilai (5).



Gambar 4.33 Rancangan antarmuka Memasukkan Nilai Ujian Peribadatan

4.3.5.5 Antarmuka ledger kemandirian

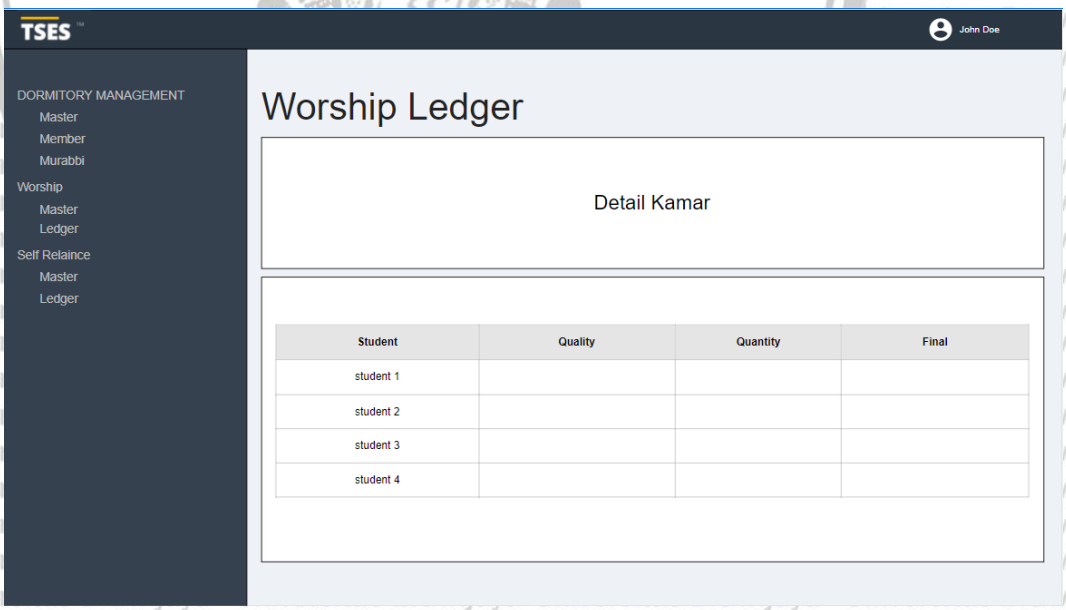
Antarmuka ledger kemandirian adalah halaman yang ditampilkan sistem ketika pengguna akan melihat nilai kemandirian santri. Antarmuka ledger kemandirian memiliki beberapa komponen di antaranya, judul halaman, detail kamar dan tabel daftar nilai kemandirian santri. Pada Gambar 4.34 adalah antarmuka ledger kemandirian.



Gambar 4.34 Rancangan antarmuka Ledger Kemandirian

4.3.5.6 Antarmuka ledger peribadatan

Antarmuka ledger peribadatan adalah halaman yang ditampilkan sistem ketika pengguna akan melihat nilai peribadatan santri. Antarmuka ledger peribadatan memiliki beberapa komponen di antaranya, judul halaman, detail kamar dan tabel daftar nilai peribadatan santri. Pada Gambar 4.35 adalah antarmuka ledger peribadatan.



Gambar 4.35 Rancangan antarmuka Ledger Peribadatan

4.4 Implementasi Sistem

Implementasi sistem merupakan proses membangun sebuah sistem berdasarkan rancangan yang sudah didefinisikan. Pada bagian ini dilakukan

implementasi sistem informasi manajemen pengembangan pengasuh. Implementasi berupa penulisan kode program, basis data, dan antarmuka.

4.4.1 Implementasi Kode Program

Implementasi kode program merupakan tahap merealisasikan hasil perancangan algoritma pada tahap perancangan sistem ke dalam kode program. Rancangan tersebut diimplementasikan menggunakan bahasa pemrograman PHP.

4.4.1.1 Kode program mengelola jurnal kemandirian

Implementasi kode program mengelola jurnal kemandirian mengacu pada perancangan *sequence diagram* mengelola jurnal kemandirian dan *class diagram* Jurnal_kemandirian. Pada aktivitas mengelola jurnal kemandirian terdapat proses menambah, mengubah dan menghapus jurnal. Tabel 4.19 adalah kode program dari fungsi "add()" yang berfungi untuk menampilkan halaman tambah jurnal kemandirian.

Tabel 4.19 Kode program fungsi: add()

No	Fungsi: add()
1	public function add() {
2	$this->mybreadcrumb->add('Add', base_url($this-
3	>data['c_name']));
4	$this->data['subtitle'] = 'add a new journal';
5	$param['filter']['pegawai'] = $this->session->userdata('id');
6	$this->data['kamar']= $this->kamar-
7	>get_data_murabbi_kamar($param);
8	$this->show('murabbi/student_journal/kemandirian_journal/add',
9	$this->data);
10	}

Fungsi "edit(id)" berfungsi untuk menampilkan halaman ubah jurnal kemandirian. Tabel 4.20 merupakan implementasi kode program dari fungsi "edit(id)".

Tabel 4.20 Kode program fungsi: edit(id)

No	Fungsi: edit(id)
1	public function edit($id){
2	$this->data['id_edit'] = $id;
3	$jurnal = $this->model->get_jurnal_by_id($id);
4	$this->data['content'] = $jurnal ? (array) $jurnal : show_404();
5	$this->data['tabel'] = $this->model-
6	>get_html_input_jurnal($jurnal->ID_RUANG, $jurnal->TANGGAL);
7	$this->show('murabbi/student_journal/kemandirian_journal/edit',
8	$this->data);
9	}

Fungsi "delete()" berfungsi untuk menghapus jurnal kemandirian. Tabel 4.21 merupakan implementasi kode program dari fungsi "delete()".

Tabel 4.21 Kode program fungsi: delete()

No	Fungsi: delete()
1	public function delete(){
2	$id = $this->input->post('id');
3	if ($id != '') {
4	$this->db->where('ID_MASTER', $id);
5	$this->db->delete('tb_kemandirian_record');
6	$this->db->where('ID', $id);

```
7       if ($this->db->delete('tb_kemandirian_master')) {
8       echo 'Success';
9       } else {
10      echo 'Error';
11      }
12  }
13  }
```

Fungsi "process()" merupakan algoritma yang digunakan untuk menyimpan atau mengubah jurnal kemandirian sesuai dengan data yang diterima. Tabel 4.22 merupakan implementasi kode program dari fungsi "process()".

Tabel 4.22 Kode program fungsi: process()

```
No  Fungsi: process()
1   public function process(){
2       $id = $this->input->post('id_edit');
3       $thn = $this->session->userdata('tahun_akademik');
4       $smt = $this->session->userdata('semester');
5       $idp = $this->session->userdata('id');
6       $mdr = $this->input->post('mandiri');
7       $ksg = $this->input->post('kosong');
8       $tgl = $this->input->post('date');
9       $kamar = $this->input->post('dormitory');
10      $id_jurnal = '';
11      $data_record = array();
12      $result = array();
13
14      $param['id'] = $id;
15      $param['filter']['ruang'] = $kamar;
16      $param['filter']['tanggal'] = date('Y/m/d', strtotime($tgl));
17      $jurnal = $this->model->get_data_jurnal($param);
18      if (sizeof($jurnal) > 0) {
19          $id_jurnal = $jurnal[0]->ID;
20          $this->db->where('ID', $id_jurnal);
21          $this->db->update('tb_kemandirian_master',
22  array('LAST_UPDATED' => date('Y-m-d H:i')));
23          $this->db->where('ID_MASTER', $id_jurnal);
24          $this->db->delete('tb_kemandirian_record');
25      } else {
26          $data = array(
27          'TANGGAL' => date('Y/m/d', strtotime($tgl)),
28          'TAHUN_AKADEMIK' => $thn,
29          'SEMESTER' => $smt,
30          'ID_PEGAWAI' => $idp,
31          'ID_RUANG' => $kamar,
32          'HARI' => date('l', strtotime($tgl)),
33          'IS_ACTIVE' => 1,
34          'DATE_CREATED' => date('Y-m-d H:i'),
35          'LAST_UPDATED' => date('Y-m-d H:i'),
36          );
37          $this->db->insert('tb_kemandirian_master', $data);
38          $id_jurnal = $this->db->insert_id();
39      }
40
41      foreach ($ksg as $siswa => $jenis) {
42          foreach ($jenis as $jen => $val) {
43          $s = 0;
44          if (isset($mdr[$siswa][$jen]) && $mdr[$siswa][$jen] == 1) {
45          $s = 1;
46          }
47          $data_record[] = array(
48          'ID_MASTER' => $id_jurnal,
49          'ID_SISWA' => $siswa,
50          'ID_JENIS' => $jen,
```

```
51          'IS_HADIR' => $s
52          );
53          }
54      }
55
56      $this->db->insert_batch('tb_kemandirian_record', $data_record);
57      $affected = $this->db->affected_rows();
58      if ($affected > 0) {
59          $result = array(
60          'status' => 'success',
61          'message' => 'Data has been saved',
62          'title' => 'Success'
63          );
64      } else {
65          $result = array(
66          'status' => 'error',
67          'message' => 'Failed to save',
68          'title' => 'Error'
69          );
70      }
71      echo json_encode($result);
72  }
```

4.4.1.2 Kode program mengelola jurnal peribadatan

Implementasi kode program mengelola jurnal peribadatan mengacu pada perancangan *sequence diagram* mengelola jurnal peribadatan dan *class diagram* Jurnal_peribadatan. Pada aktivitas mengelola jurnal peribadatan terdapat proses menambah, mengubah dan menghapus jurnal. Tabel 4.23 adalah kode program dari fungsi "add()" yang berfungi untuk menampilkan halaman tambah jurnal peribadatan.

Tabel 4.23 Kode program fungsi: add()

No	Fungsi: add()
1	public function add() {
2	$this->data['subtitle'] = 'add a new journal';
3	$this->mybreadcrumb->add('Add', base_url($this-
4	>data['c_name']));
5	$param['filter']['pegawai'] = $this->session->userdata('id');
6	$this->data['kamar'] = $this->kamar-
7	>get_data_murabbi_kamar($param);
8	$this->show('murabbi/student_journal/peribadatan_journal/add',
9	$this->data);
10	}

Fungsi "edit(id)" berfungsi untuk menampilkan halaman ubah jurnal peribadatan. Tabel 4.24 merupakan implementasi kode program dari fungsi "edit(id)".

Tabel 4.24 Kode program fungsi: edit(id)

No	Fungsi: edit(id)
1	public function edit($id){
2	$this->data['id_edit'] = $id;
3	$jurnal = $this->model->get_jurnal_by_id($id);
4	$this->data['content'] = $jurnal ? (array) $jurnal : show_404();
5	$this->data['tabel'] = $this->model-
6	>get_html_input_jurnal($jurnal->ID_RUANG, $jurnal->TANGGAL);
7	$this->show('murabbi/student_journal/peribadatan_journal/edit',
8	$this->data);
9	}
10	

Fungsi “delete()” berfungsi untuk menghapus jurnal peribadatan. Tabel 4.25 merupakan implementasi kode program dari fungsi “delete()”.

Tabel 4.25 Kode program fungsi: delete()

No	Fungsi: delete()
1	public function delete(){
2	$id = $this->input->post('id');
3	if ($id != '') {
4	$this->db->where('ID_MASTER', $id);
5	$this->db->delete('tb_peribadatan_record');
6	$this->db->where('ID', $id);
7	if ($this->db->delete('tb_peribadatan_master')) {
8	echo 'Success';
9	} else {
10	echo 'Error';
11	}
12	}
13	}

Fungsi “process()” merupakan algoritma yang digunakan untuk menyimpan atau mengubah jurnal peribadatan sesuai dengan data yang diterima. Tabel 4.26merupakan implementasi kode program dari fungsi “process()”.

Tabel 4.26 Kode program fungsi: process()

No	Fungsi: process()
1	public function process()
2	{
3	$id = $this->input->post('id_edit');
4	$thn = $this->session->userdata('tahun_akademik');
5	$smt = $this->session->userdata('semester');
6	$idp = $this->session->userdata('id');
7	$mdr = $this->input->post('ibadah');
8	$ksg = $this->input->post('kosong');
9	$tgl = $this->input->post('date');
10	$kamar = $this->input->post('dormitory');
11	$id_jurnal = '';
12	$data_record = array();
13	$result = array();
14	
15	$param['id'] = $id;
16	$param['filter']['ruang'] = $kamar;
17	$param['filter']['tanggal'] = date('Y/m/d', strtotime($tgl));
18	$jurnal = $this->model->get_data_jurnal($param);
19	if (sizeof($jurnal) > 0) {
20	$id_jurnal = $jurnal[0]->ID;
21	$this->db->where('ID', $id_jurnal);
22	$this->db->update('tb_peribadatan_master',
23	array('LAST_UPDATED' => date('Y-m-d H:i')));
24	$this->db->where('ID_MASTER', $id_jurnal);
25	$this->db->delete('tb_peribadatan_record');
26	} else {
27	$data = array(
28	'TANGGAL' => date('Y/m/d', strtotime($tgl)),
29	'TAHUN_AKADEMIK' => $thn,
30	'SEMESTER' => $smt,
31	'ID_PEGAWAI' => $idp,
32	'ID_RUANG' => $kamar,
33	'IS_ACTIVE' => 1,
34	'DATE_CREATED' => date('Y-m-d H:i'),
35	'LAST_UPDATED' => date('Y-m-d H:i'),
36	);
37	$this->db->insert('tb_peribadatan_master', $data);

```
38          $id_jurnal = $this->db->insert_id();
39      }
40
41      foreach ($ksg as $siswa => $jenis) {
42          foreach ($jenis as $jen => $val) {
43          $s = 0;
44          if (isset($mdr[$siswa][$jen]) && $mdr[$siswa][$jen] == 1) {
45           $s = 1;
46          }
47          $data_record[] = array(
48          'ID_MASTER' => $id_jurnal,
49          'ID_SISWA' => $siswa,
50          'ID_JENIS' => $jen,
51          'IS_HADIR' => $s
52          );
53          }
54      }
55
56      $this->db->insert_batch('tb_peribadatan_record', $data_record);
57      $affected = $this->db->affected_rows();
58      if ($affected > 0) {
59          $result = array(
60          'status' => 'success',
61          'message' => 'Data has been saved',
62          'title' => 'Success'
63          );
64      } else {
65          $result = array(
66          'status' => 'error',
67          'message' => 'Failed to save',
68          'title' => 'Error'
69          );
70      }
71      echo json_encode($result);
72  }
```

4.4.1.3 Kode program mengelola nilai ujian kemandirian

Implementasi kode program mengelola nilai ujian kemandirian mengacu pada perancangan *sequence diagram* mengelola nilai ujian kemandirian dan *class diagram* Ujian_kemandirian. Pada aktivitas mengelola nilai ujian kemandirian terdapat proses menambah nilai ujian kemandirian. Tabel 4.27 adalah kode program dari fungsi "detail(id)" yang berfungi untuk menampilkan halaman mengisi nilai ujian kemandirian suatu kamar.

Tabel 4.27 Kode program fungsi: detail(id)

No	Fungsi: detail(id)
1	public function detail($id){
2	$this->data['subtitle'] = 'input exam score';
3	$this->mybreadcrumb->add('Detail', base_url($this-
4	>data['c_name']));
5	$kamar = $this->kamar->get_murabbi_kamar_by_id($id);
6	$this->data['content'] = $kamar ? (array) $kamar : show_404();
7	$this->show('murabbi/student_journal/kemandirian_exam/detail',
8	$this->data);
9	}

Fungsi "process()" merupakan algoritma yang digunakan untuk menyimpan atau mengubah nilai ujian kemandirian sesuai dengan data yang diterima. Tabel 4.28merupakan implementasi kode program dari fungsi "process()".

Tabel 4.28 Kode program fungsi: process()

No	Fungsi: process()
1	public function process(){
2	$form = $this->input->post('form');
3	$nilai = $this->input->post('nilai');
4	$aspek = $this->input->post('aspek');
5	$id_insert = '';
6	
7	$result = array();
8	$cek = array(
9	'TAHUN_AKADEMIK' => $form['tahun'],
10	'SEMESTER' => $form['semester'],
11	'ID_RUANG' => $form['kamar'],
12	);
13	$data = array(
14	'TAHUN_AKADEMIK' => $form['tahun'],
15	'SEMESTER' => $form['semester'],
16	'ID_PEGAWAI' => $this->session->userdata('id'),
17	'ID_RUANG' => $form['kamar'],
18	'LEVEL' => $form['level'],
19	'DATE_CREATED' => date('Y-m-d H:i'),
20	'LAST_UPDATED' => date('Y-m-d H:i'),
21	);
22	$this->db->where($cek);
23	$cektgl = $this->db->get('tb_kemandirian_ujian_master');
24	if ($cektgl->num_rows() > 0) {
25	$id_insert = $cektgl->row()->ID;
26	$this->db->where('ID', $id_insert);
27	$this->db->update('tb_kemandirian_ujian_master',
28	array('LEVEL' => $form['level'], 'LAST_UPDATED' => date('Y-m-d
29	H:i')));
30	$this->db->where('ID_MASTER', $id_insert);
31	$this->db->delete('tb_kemandirian_ujian_record');
32	} else {
33	$this->db->insert('tb_kemandirian_ujian_master', $data);
34	$id_insert = $this->db->insert_id();
35	}
36	foreach ($nilai as $siswa => $value) {
37	foreach ($value as $row => $val) {
38	$data2[] = array(
39	'ID_MASTER' => $id_insert,
40	'ID_SISWA' => $siswa,
41	'ASPEK' => $aspek[$siswa][$row],
42	'NILAI' => $val,
43	);
44	}
45	}
46	$this->db->insert_batch('tb_kemandirian_ujian_record', $data2);
47	$affected = $this->db->affected_rows();
48	if ($affected > 0) {
49	$result = array(
50	'status' => 'success',
51	'message' => 'Data has been saved',
52	'title' => 'Success'
53	);
54	} else {
55	$result = array(
56	'status' => 'error',
57	'message' => 'Failed to save',
58	'title' => 'Error'
59	);
60	}
61	echo json_encode($result);
62	}

4.4.1.4 Kode program mengelola nilai ujian peribadatan

Implementasi kode program mengelola nilai ujian peribadatan mengacu pada perancangan *sequence diagram* mengelola nilai ujian peribadatan dan *class diagram* Ujian_peribadatan. Pada aktivitas mengelola nilai ujian peribadatan terdapat proses menambah nilai ujian peribadatan. Tabel 4.29 adalah kode program dari fungsi "detail(id)" yang berfungi untuk menampilkan halaman mengisi nilai ujian kemandirian suatu kamar.

Tabel 4.29 Kode program fungsi: detail(id)

No	Fungsi: detail(id)
1	public function detail($id){
2	$this->data['subtitle'] = 'input exam score';
3	$this->mybreadcrumb->add('Detail', base_url($this-
4	>data['c_name']));
5	$kamar = $this->kamar->get_murabbi_kamar_by_id($id);
6	$this->data['content'] = $kamar ? (array) $kamar : show_404();
7	$this->show('murabbi/student_journal/peribadatan_exam/detail',
8	$this->data);
9	}

Fungsi "process()" merupakan algoritma yang digunakan untuk menyimpan atau mengubah nilai ujian kemandirian sesuai dengan data yang diterima. Tabel 4.30 merupakan implementasi kode program dari fungsi "process()".

Tabel 4.30 Kode program fungsi: process()

No	Fungsi: process()
1	public function process(){
2	$form = $this->input->post('form');
3	$nilai = $this->input->post('nilai');
4	$aspek = $this->input->post('aspek');
5	$id_insert = '';
6	
7	$result = array();
8	$cek = array(
9	'TAHUN_AKADEMIK' => $form['tahun'],
10	'SEMESTER' => $form['semester'],
11	'ID_RUANG' => $form['kamar'],
12	);
13	$data = array(
14	'TAHUN_AKADEMIK' => $form['tahun'],
15	'SEMESTER' => $form['semester'],
16	'ID_PEGAWAI' => $this->session->userdata('id'),
17	'ID_RUANG' => $form['kamar'],
18	'DATE_CREATED' => date('Y-m-d H:i'),
19	'LAST_UPDATED' => date('Y-m-d H:i'),
20	);
21	$this->db->where($cek);
22	$cektgl = $this->db->get('tb_peribadatan_ujian_master');
23	if ($cektgl->num_rows() > 0) {
24	$id_insert = $cektgl->row()->ID;
25	$this->db->where('ID', $id_insert);
26	$this->db->update('tb_peribadatan_ujian_master',
27	array('LAST_UPDATED' => date('Y-m-d H:i')));
28	$this->db->where('ID_MASTER', $id_insert);
29	$this->db->delete('tb_peribadatan_ujian_record');
30	} else {
31	$this->db->insert('tb_peribadatan_ujian_master', $data);
32	$id_insert = $this->db->insert_id();
33	}
34	foreach ($nilai as $siswa => $value) {

```
35          foreach ($value as $row => $val) {
36              $data2[] = array(
37              'ID_MASTER' => $id_insert,
38              'ID_SISWA' => $siswa,
39              'ASPEK' => $aspek[$siswa][$row],
40              'NILAI' => $val,
41              );
42          }
43           }
44          $this->db->insert_batch('tb_peribadatan_ujian_record',
45  $data2);
46          $affected = $this->db->affected_rows();
47          if ($affected > 0) {
48          $result = array(
49              'status' => 'success',
50              'message' => 'Data has been saved',
51              'title' => 'Success'
52          );
53          } else {
54          $result = array(
55              'status' => 'error',
56              'message' => 'Failed to save',
57              'title' => 'Error'
58          );
59           }
60      echo json_encode($result);
61       }
```

4.4.1.5 Kode program melihat ledger kemandirian

Implementasi kode program mengelola nilai ujian peribadatan mengacu pada perancangan *sequence diagram* melihat ledger kemandirian dan *class diagram* Ledger_kemandirian. Pada aktivitas melihat ledger kemandirian terdapat proses menampilkan ledger kemandirian. Tabel 4.31 adalah kode program dari fungsi "detail(id)" yang berfungi untuk menampilkan halaman detail ledger kemandirian suatu kamar.

Tabel 4.31 Kode program fungsi: detail(id)

No	Fungsi: detail(id)
1	`public function detail($id) {`
2	`    $this->data['subtitle'] = 'Detail - ' . $this->data['title'];`
3	`    $this->mybreadcrumb->add('Detail', base_url($this-`
4	`>data['c_name']));`
5	`    $kamar = $this->kamar->get_murabbi_kamar_by_id($id);`
6	`    $this->data['content'] = $kamar ? (array) $kamar : show_404();`
7	
8	`    $param1['filter']['tahun'] = $kamar->TAHUN_AKADEMIK;`
9	`    $param1['filter']['semester'] = $kamar->SEMESTER;`
10	`    $param1['filter']['kamar'] = $kamar->ID_RUANG;`
11	`    $param2['tahun'] = $kamar->TAHUN_AKADEMIK;`
12	`    $param2['semester'] = $kamar->SEMESTER;`
13	`    $param2['murabbi_kamar'] = $kamar->ID;`
14	`    $this->data['ledger']['kuantitas'] = $this->model-`
15	`>geta_data_kuantitas($param2);`
16	`    $this->data['ledger']['kualitas'] = $this->model-`
17	`>geta_data_kualitas($param2);`
18	
19	`    $this->data['santri'] = $this->santri-`
20	`>get_data_santri($param1);;`
21	`    $this->data['aspek_kuantitas'] = $this->model-`
22	`>get_aspek_kuantitas();`
23	

```
24      $this->data['aspek_kualitas']          =          $this->model-
25  >get_aspek_kualitas();
26      $this->show('student_journal/kemandirian_ledger/detail',  $this-
27  >data);
28      }
```

4.4.1.6 Kode program melihat ledger peribadatan

Implementasi kode program mengelola nilai ujian peribadatan mengacu pada perancangan *sequence diagram* melihat ledger kemandirian dan *class diagram* Ledger_peribadatan. Pada aktivitas melihat ledger peribadatan terdapat proses menampilkan ledger peribadatan. Tabel 4.32 adalah kode program dari fungsi "detail(id)" yang berfungi untuk menampilkan halaman detail ledger peribadatan suatu kamar.

Tabel 4.32 Kode program fungsi: detail(id)

```
No  Fungsi: detail(id)
1   public function detail($id){
2       $this->data['subtitle'] = 'Detail - ' . $this->data['title'];
3       $this->mybreadcrumb->add('Detail',                base_url($this-
4   >data['c_name']));
5       $kamar = $this->kamar->get_murabbi_kamar_by_id($id);
6       $this->data['content'] = $kamar ? (array) $kamar : show_404();
7
8       $param1['filter']['tahun'] = $kamar->TAHUN_AKADEMIK;
9       $param1['filter']['semester'] = $kamar->SEMESTER;
10      $param1['filter']['kamar'] = $kamar->ID_RUANG;
11      $param2['tahun'] = $kamar->TAHUN_AKADEMIK;
12      $param2['semester'] = $kamar->SEMESTER;
13      $param2['murabbi_kamar'] = $kamar->ID;
14
15      $this->data['santri'] = $this->santri->get_data_santri($param1);
16      $this->data['ledger']['kuantitas']        =        $this->model-
17  >geta_data_kuantitas($param2);
18      $this->data['ledger']['kualitas']         =        $this->model-
19  >geta_data_kualitas($param2);
20
21      $this->data['aspek_kuantitas']           =         $this->model-
22  >get_aspek_kuantitas();
23      $this->data['aspek_kualitas']            =         $this->model-
24  >get_aspek_kualitas();
25      $this->data['aspek_kualitas'] = array('');
26      $this->show('student_journal/peribadatan_ledger/detail',  $this-
27  >data);
28      }
```

4.4.2 Implementasi Basis Data

Implementasi basis data merupakan tahap merealisasikan hasil perancangan pada tahap pemetaan *class* ke model relasional. Implementasi basis data membuat dan memodifikasi objek basis data seperti tabel, indeks, dan batasan (*constrain*). Tabel 4.33 merupakan pernyataan *data definition language (DDL)* untuk mendefinisikan struktur data.

Tabel 4.33 Pernyataan DDL

```
No  DDL TABEL
1   CREATE TABLE `m_gedung` (
2     `ID` int(11) NOT NULL AUTO_INCREMENT,
```

```
  `NAMA_GEDUNG` varchar(255) NULL,
  `KAMPUS` smallint(1) NULL,
  `IS_ACTIVE` smallint(1) NULL DEFAULT 1,
  PRIMARY KEY (`ID`)
);
CREATE TABLE `m_jenis_kemandirian` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `ACTIVITY` varchar(255) NULL,
  `KATEGORI` varchar(255) NULL,
  `HARI` varchar(255) NOT NULL,
  `ALIAS` varchar(255) NULL,
  PRIMARY KEY (`ID`)
);

CREATE TABLE `m_jenis_peribadatan` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `KATEGORI` smallint(2) NULL,
  `ACTIVITY` varchar(255) NULL,
  `SINGKAT` varchar(255) NULL,
  PRIMARY KEY (`ID`)
);

CREATE TABLE `m_pegawai` (
  `ID_PEGAWAI` int(11) NOT NULL AUTO_INCREMENT,
  `USERNAME` varchar(40) NOT NULL,
  `PASSWORD` varchar(255) NOT NULL,
  `EMAIL` varchar(225) NULL,
  `HAK_AKSES` int(11) NULL DEFAULT 2,
  `NIY` varchar(30) NOT NULL,
  `NAMA` varchar(100) NOT NULL,
  `TEMPAT_LAHIR` varchar(60) NULL,
  `TANGGAL_LAHIR` date NULL,
  `JENIS_KELAMIN` enum('L','P') NULL,
  `JABATAN_STRUKTURAL` varchar(60) NULL,
  `JABATAN_FUNGSIONAL` varchar(60) NULL,
  `TANGGAL_AKTIF` date NULL,
  `IS_ACTIVE` tinyint(1) NULL DEFAULT 1,
  `FOTO` varchar(255) 'assets/icon/USER.png',
  PRIMARY KEY (`ID_PEGAWAI`),
  UNIQUE INDEX `USERNAME`(`USERNAME`)
);

CREATE TABLE `m_ruang` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `KODE` varchar(20) NULL,
  `NAMA_RUANG` varchar(80) NULL,
  `NAMA_ARAB` varchar(255) NULL,
  `TIPE` smallint(2) NULL COMMENT '1 = kamar santri| 2 = kamar
murabbi| 3 = kelas | 4 = kantor | 5 = public area',
  `IS_ACTIVE` smallint(1) NULL DEFAULT 1,
  `ID_GEDUNG` int(11) NULL,
  PRIMARY KEY (`ID`),
  INDEX `NAMA_RUANG`(`NAMA_RUANG`)
);

CREATE TABLE `m_siswa` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `NIS` varchar(60) NOT NULL,
  `NIK` char(16) NULL,
  `NISN` char(10) NULL,
  `PASSWORD` varchar(255) NOT NULL,
  `NAMA_LENGKAP` varchar(100) NOT NULL,
  `JENJANG` enum('SMP','SMA') NOT NULL,
  `AGAMA` varchar(20) NULL,
```

```
68      `JENIS_KELAMIN` enum('L','P') NULL,
69      `TEMPAT_LAHIR` varchar(50) NULL,
70      `TANGGAL_LAHIR` date NULL,
71      `ALAMAT` varchar(255) NULL,
72      `KEWARGANEGARAAN` varchar(40) NULL,
73      `FOTO` varchar(255) NOT NULL DEFAULT 'assets/icon/USER.png',
74      `IS_ACTIVE` tinyint(1) NULL DEFAULT 1,
75      `TAHUN_MASUK` smallint(4) NULL,
76      `TAHUN_KELUAR` smallint(4) NULL,
77      `STATUS` smallint(1) NULL DEFAULT 1,
78      PRIMARY KEY (`ID`),
79      INDEX `nama`(`NAMA_LENGKAP`)
80    );
81
82    CREATE TABLE `tb_dormitory_murabbi`  (
83      `ID` int(11) NOT NULL AUTO_INCREMENT,
84      `ID_PEGAWAI` int(11) NOT NULL,
85      `ID_RUANG` int(11) NOT NULL,
86      `TAHUN_AKADEMIK` varchar(20) NULL,
87      `SEMESTER` smallint(1) NULL,
88      `IS_ACTIVE` smallint(1) NOT NULL DEFAULT 1,
89      PRIMARY KEY (`ID`),
90      INDEX `TAHUN_AKADEMIK`(`TAHUN_AKADEMIK`, `SEMESTER`)
91    );
92
93    CREATE TABLE `tb_kemandirian_master`  (
94      `ID` int(11) NOT NULL AUTO_INCREMENT,
95      `TANGGAL` date NOT NULL,
96      `TAHUN_AKADEMIK` varchar(20) NOT NULL,
97      `SEMESTER` smallint(1) NOT NULL,
98      `ID_PEGAWAI` int(11) NOT NULL,
99      `HARI` varchar(25) NOT NULL,
100     `IS_ACTIVE` smallint(1) NOT NULL,
101     `ID_RUANG` int(11) NULL,
102     `DATE_CREATED` datetime(0) NULL,
103     `LAST_UPDATED` datetime(0) NULL,
104     PRIMARY KEY (`ID`),
105     INDEX `ID_PEGAWAI`(`ID_PEGAWAI`)
106   );
107
108   CREATE TABLE `tb_kemandirian_record`  (
109     `ID` int(11) NOT NULL AUTO_INCREMENT,
110     `ID_MASTER` int(11) NOT NULL,
111     `ID_SISWA` int(11) NOT NULL,
112     `ID_JENIS` int(11) NOT NULL,
113     `IS_HADIR` smallint(1) NOT NULL,
114     PRIMARY KEY (`ID`),
115     INDEX `ID_SISWA`(`ID_SISWA`),
116     INDEX `ID_JENIS`(`ID_JENIS`),
117     INDEX `ID_MASTER`(`ID_MASTER`)
118   );
119
120   CREATE TABLE `tb_kemandirian_ujian_master`  (
121     `ID` int(11) NOT NULL AUTO_INCREMENT,
122     `ID_RUANG` int(11) NULL,
123     `ID_PEGAWAI` int(11) NULL,
124     `TIPE` enum('UTS','UAS') NULL,
125     `LEVEL` enum('SMP','SMA') NULL,
126     `TAHUN_AKADEMIK` varchar(20) NULL,
127     `SEMESTER` smallint(1) NULL,
128     `DATE_CREATED` datetime(0) NULL,
129     `LAST_UPDATED` datetime(0) NULL,
130     PRIMARY KEY (`ID`),
131     INDEX `TAHUN_AKADEMIK`(`TAHUN_AKADEMIK`, `SEMESTER`)
132   );
```

```
133
134  CREATE TABLE `tb_kemandirian_ujian_record` (
135    `ID` int(11) NOT NULL AUTO_INCREMENT,
136    `ID_MASTER` int(11) NULL,
137    `ID_SISWA` int(11) NULL,
138    `ASPEK` varchar(255) NULL,
139    `NILAI` decimal(10, 0) NULL,
140    PRIMARY KEY (`ID`)
141  );
142
143  CREATE TABLE `tb_peribadatan_master` (
144    `ID` int(11) NOT NULL AUTO_INCREMENT,
145    `TANGGAL` date NULL,
146    `TAHUN_AKADEMIK` varchar(20) NULL,
147    `SEMESTER` smallint(1) NULL,
148    `ID_PEGAWAI` int(11) NULL,
149    `ID_KATEGORI` int(11) NULL,
150    `IS_ACTIVE` smallint(1) NULL,
151    `ID_RUANG` int(11) NULL,
152    `DATE_CREATED` datetime(0) NULL,
153    `LAST_UPDATED` datetime(0) NULL,
154    PRIMARY KEY (`ID`),
155    INDEX `TAHUN_AKADEMIK`(`TAHUN_AKADEMIK`, `SEMESTER`)
156  );
157
158  CREATE TABLE `tb_peribadatan_record` (
159    `ID` int(11) NOT NULL AUTO_INCREMENT,
160    `ID_MASTER` int(11) NULL,
161    `ID_SISWA` int(11) NULL,
162    `ID_JENIS` int(11) NULL,
163    `IS_HADIR` smallint(1) NULL COMMENT '1==HADIR,0==TIDAK HARIR',
164    PRIMARY KEY (`ID`)
165  );
166
167  CREATE TABLE `tb_peribadatan_ujian_master` (
168    `ID` int(11) NOT NULL AUTO_INCREMENT,
169    `ID_RUANG` int(11) NULL,
170    `ID_PEGAWAI` int(11) NULL,
171    `TIPE` enum('UTS','UAS') NULL,
172    `TAHUN_AKADEMIK` varchar(20) NULL,
173    `SEMESTER` smallint(1) NULL,
174    `DATE_CREATED` datetime(0) NULL,
175    `LAST_UPDATED` datetime(0) NULL,
176    PRIMARY KEY (`ID`),
177    INDEX `TAHUN_AKADEMIK`(`TAHUN_AKADEMIK`, `SEMESTER`)
178  );
179
180  CREATE TABLE `tb_peribadatan_ujian_record` (
181    `ID` int(11) NOT NULL AUTO_INCREMENT,
182    `ID_MASTER` int(11) NULL,
183    `ID_SISWA` int(11) NULL,
184    `ASPEK` varchar(255) NULL,
185    `NILAI` decimal(10, 0) NULL,
186    PRIMARY KEY (`ID`)
187  );
188
189  CREATE TABLE `tb_siswa_kamar` (
190    `ID` int(11) NOT NULL AUTO_INCREMENT,
191    `ID_RUANG` int(11) NULL,
192    `ID_SISWA` int(11) NULL,
193    `TAHUN_AKADEMIK` varchar(10) NULL,
194    `SEMESTER` smallint(1) NULL,
195    PRIMARY KEY (`ID`),
196    UNIQUE INDEX `ID_SISWA`(`ID_SISWA`,
197  `TAHUN_AKADEMIK`,`SEMESTER`),
```

```
198 );
199
200 ALTER TABLE `m_ruang` ADD FOREIGN KEY (`ID_GEDUNG`) REFERENCES
201 `m_gedung` (`ID`);
202 ALTER TABLE `tb_dormitory_murabbi` ADD FOREIGN KEY
203 (`ID_PEGAWAI`) REFERENCES `m_pegawai` (`ID_PEGAWAI`);
204 ALTER TABLE `tb_dormitory_murabbi` ADD FOREIGN KEY (`ID_RUANG`)
205 REFERENCES `m_ruang` (`ID`);
206 ALTER TABLE `tb_kemandirian_master` ADD FOREIGN KEY
207 (`ID_PEGAWAI`) REFERENCES `m_pegawai` (`ID_PEGAWAI`);
208 ALTER TABLE `tb_kemandirian_record` ADD FOREIGN KEY (`ID_SISWA`)
209 REFERENCES `m_siswa` (`ID`);
210 ALTER TABLE `tb_kemandirian_record` ADD FOREIGN KEY (`ID_JENIS`)
211 REFERENCES `m_jenis_kemandirian` (`ID`);
212 ALTER TABLE `tb_kemandirian_record` ADD FOREIGN KEY
213 (`ID_MASTER`) REFERENCES `tb_kemandirian_master` (`ID`);
214 ALTER TABLE `tb_kemandirian_ujian_master` ADD FOREIGN KEY
215 (`ID_RUANG`) REFERENCES `m_ruang` (`ID`);
216 ALTER TABLE `tb_kemandirian_ujian_master` ADD FOREIGN KEY
217 (`ID_PEGAWAI`) REFERENCES `m_pegawai` (`ID_PEGAWAI`);
218 ALTER TABLE `tb_kemandirian_ujian_record` ADD FOREIGN KEY
219 (`ID_MASTER`) REFERENCES `tb_kemandirian_ujian_master` (`ID`);
220 ALTER TABLE `tb_kemandirian_ujian_record` ADD FOREIGN KEY
221 (`ID_SISWA`) REFERENCES `m_siswa` (`ID`);
222 ALTER TABLE `tb_peribadatan_master` ADD FOREIGN KEY (`ID_RUANG`)
223 REFERENCES `m_ruang` (`ID`);
224 ALTER TABLE `tb_peribadatan_master` ADD FOREIGN KEY
225 (`ID_PEGAWAI`) REFERENCES `m_pegawai` (`ID_PEGAWAI`);
226 ALTER TABLE `tb_peribadatan_record` ADD FOREIGN KEY (`ID_SISWA`)
227 REFERENCES `m_siswa` (`ID`);
228 ALTER TABLE `tb_peribadatan_record` ADD FOREIGN KEY
229 (`ID_MASTER`) REFERENCES `tb_peribadatan_master` (`ID`);
230 ALTER TABLE `tb_peribadatan_record` ADD FOREIGN KEY (`ID_JENIS`)
231 REFERENCES `m_jenis_peribadatan` (`ID`);
232 ALTER TABLE `tb_peribadatan_ujian_master` ADD FOREIGN KEY
233 (`ID_RUANG`) REFERENCES `m_ruang` (`ID`);
234 ALTER TABLE `tb_peribadatan_ujian_master` ADD FOREIGN KEY
235 (`ID_PEGAWAI`) REFERENCES `m_pegawai` (`ID_PEGAWAI`);
236 ALTER TABLE `tb_peribadatan_ujian_record` ADD FOREIGN KEY
237 (`ID_MASTER`) REFERENCES `tb_peribadatan_ujian_master` (`ID`);
238 ALTER TABLE `tb_peribadatan_ujian_record` ADD FOREIGN KEY
239 (`ID_SISWA`) REFERENCES `m_siswa` (`ID`);
240 ALTER TABLE `tb_siswa_kamar` ADD FOREIGN KEY (`ID_RUANG`)
241 REFERENCES `m_ruang` (`ID`);
242 ALTER TABLE `tb_siswa_kamar` ADD FOREIGN KEY (`ID_SISWA`)
243 REFERENCES `m_siswa` (`ID`);
```

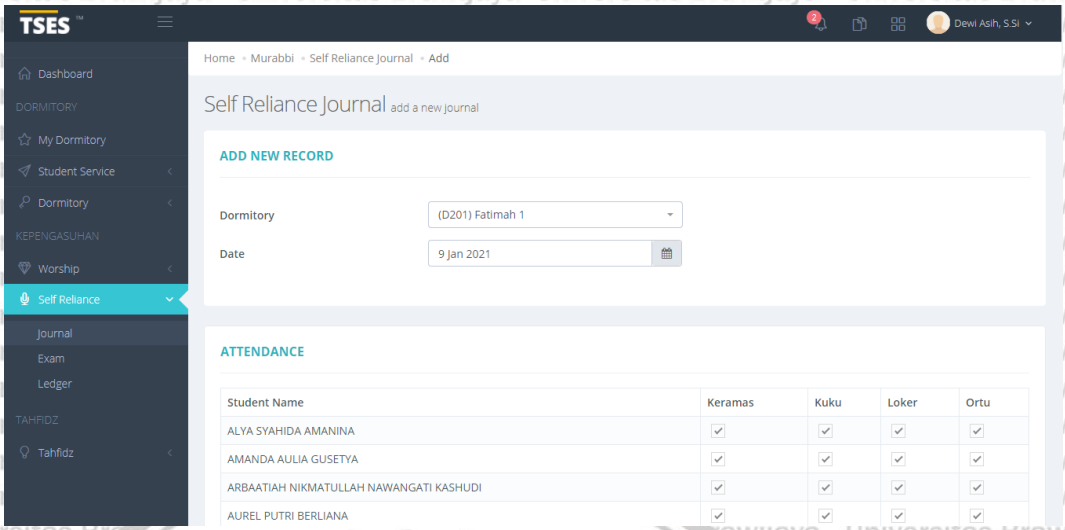
4.4.3 Implementasi Antarmuka

Implementasi antarmuka merupakan tahap merealisasikan hasil perancangan antarmuka menjadi tampilan *website* yang dapat dilihat langsung oleh pengguna. Implementasi antarmuka menggunakan *template admin dashboard* Metronic agar mempermudah implementasi dan menjaga konsistensi tampilan. Metronic menyediakan berbagai macam komponen tampilan yang dapat digunakan untuk membangun antarmuka sebuah *website*.

4.4.3.1 Implementasi antarmuka mengelola jurnal kemandirian

Gambar 4.36 menunjukkan implementasi antarmuka mengelola jurnal kemandirian saat melakukan tambah jurnal. Implementasi antarmuka

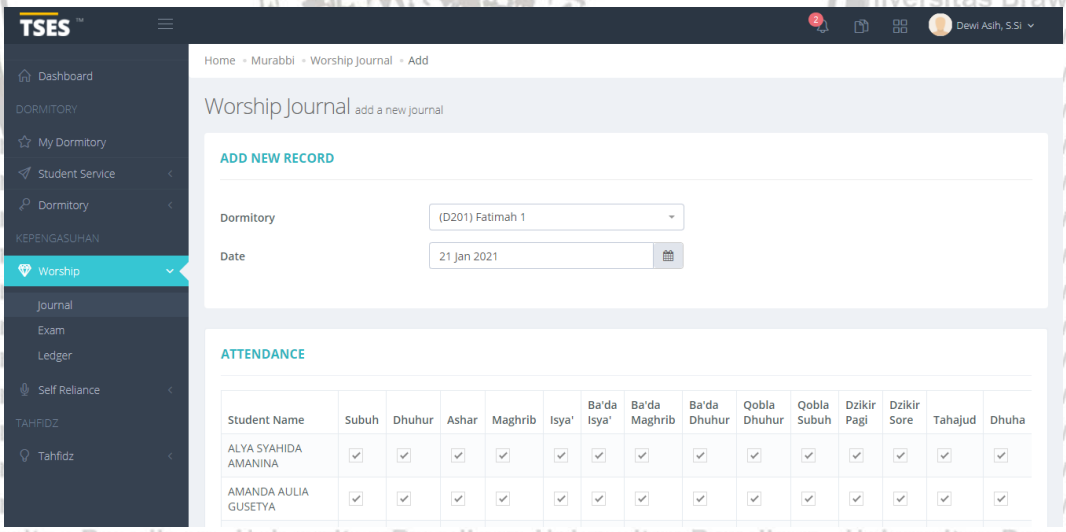
berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka mengelola jurnal kemandirian.



Gambar 4.36 Implementasi antarmuka mengelola jurnal kemandirian

4.4.3.2 Implementasi antarmuka mengelola jurnal peribadatan

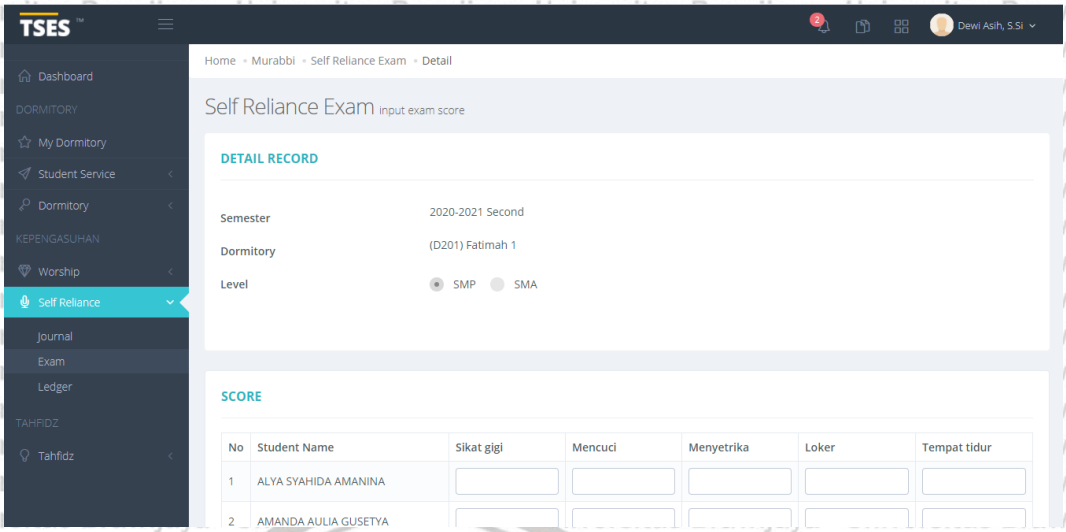
Gambar 4.37 menunjukkan implementasi antarmuka mengelola jurnal peribadatan saat melakukan tambah jurnal. Implementasi antarmuka berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka mengelola jurnal peribadatan.



Gambar 4.37 Implementasi antarmuka mengelola jurnal peribadatan

4.4.3.3 Implementasi antarmuka mengelola ujian kemandirian

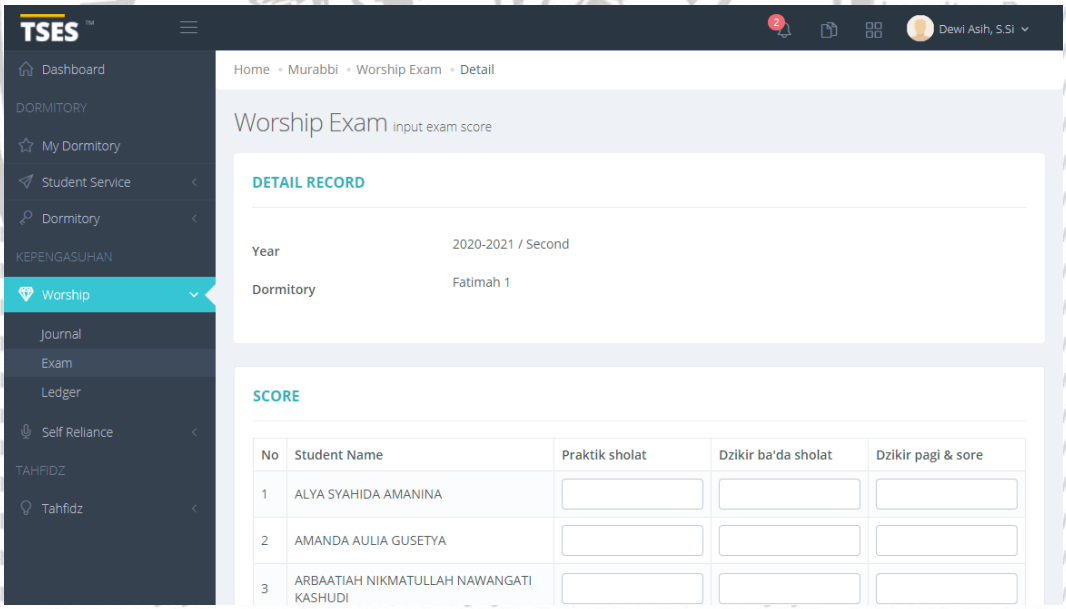
Gambar 4.38 menunjukkan implementasi antarmuka mengelola ujian kemandirian saat memasukkan nilai. Implementasi antarmuka berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka mengelola ujian kemandirian.



Gambar 4.38 Implementasi antarmuka mengelola ujian kemandirian

4.4.3.4 Implementasi antarmuka mengelola ujian peribadatan

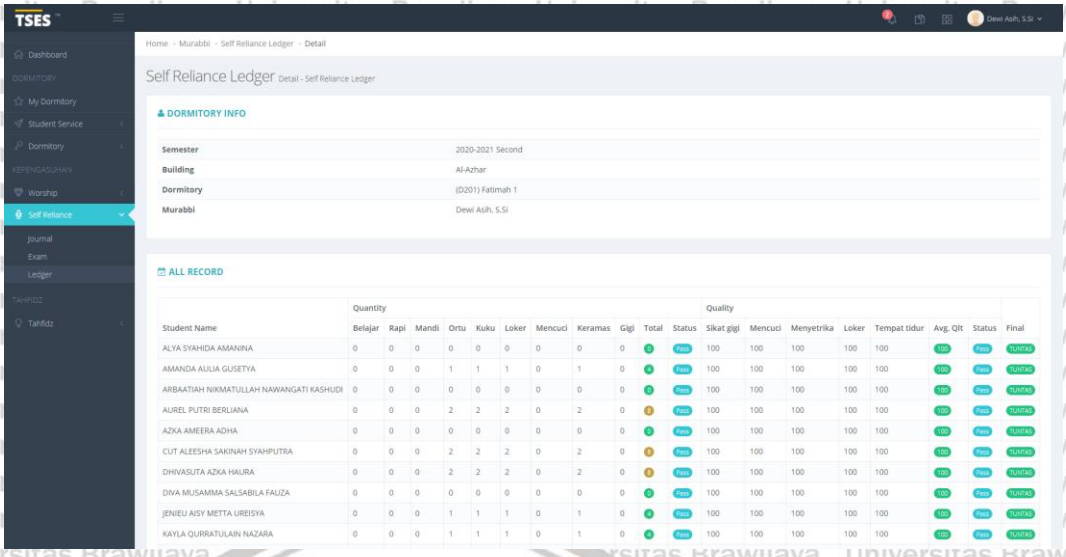
Gambar 4.39 menunjukkan implementasi antarmuka mengelola ujian peribadatan saat memasukkan nilai. Implementasi antarmuka berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka mengelola ujian peribadatan.



Gambar 4.39 Implementasi antarmuka mengelola ujian peribadatan

4.4.3.5 Implementasi antarmuka ledger kemandirian

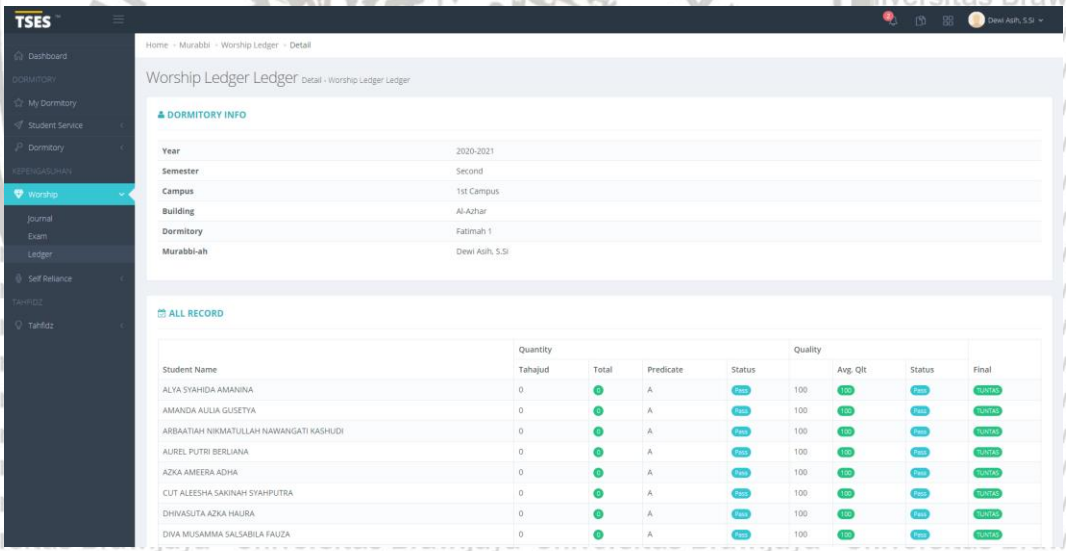
Gambar 4.40 menunjukkan implementasi antarmuka ledger kemandirian. Implementasi antarmuka berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka ledger kemandirian.



Gambar 4.40 Implementasi antarmuka ledger kemandirian

4.4.3.6 Implementasi antarmuka ledger peribadatan

Gambar 4.41 menunjukkan implementasi antarmuka ledger peribadatan. Implementasi antarmuka berdasarkan perancangan yang sudah dibuat pada bagian perancangan antarmuka ledger peribadatan.



Gambar 4.41 Implementasi antarmuka ledger peribadatan

4.5 Pengujian Sistem

Pengujian sistem merupakan tahapan yang dilakukan untuk mengetahui kesalahan yang terdapat pada sistem informasi yang dibangun. Tujuan dilakukannya *validation testing* adalah untuk mengetahui apakah hasil implementasi telah memenuhi fungsionalitasnya atau tidak dan mengetahui apakah terdapat cacat ataupun eror.

4.5.1 *Validation Testing*

Pengujian validitas dilakukan dengan menjalankan serangkaian kasus uji. Kasus uji yang akan dijalankan dibuat berdasarkan *use case*. Hasil pengujian kemudian dianalisis dengan membandingkannya dengan ekspektasi pengujian. Setiap pengujian diberi kode pengujian dengan format VT-01, VT-02, dst.

4.5.1.1 *Validation testing login*

Tabel 4.34 menunjukkan pengujian validasi pada fungsional *login*. Terdapat dua kasus pengujian pada modul *login*. Pertama *login* dengan *username* dan *password* yang valid. Kedua *login* dengan *username* atau *password* yang tidak valid.

Tabel 4.34 *Test Case Login*

Modul	: Fungsional login
Deskripsi	: Menguji jalannya fungsi login yang berfungsi untuk melakukan autentikasi dan membagi hak akses kepada pengguna sistem
Traceability	: Kebutuhan Fungsional(KF-01), *Use Case Diagram Login, Activity Diagram Login*

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-01	Kasus 1: Memasukkan *username* dan *password* yang valid	Memasukkan *username* → memasukkan *password* → menekan tombol *login*	*Username* : 15062025 *Password* : 15062025	Pengguna berhasil masuk ke sistem dan mendapat modul sesuai hak aksesnya	Pengguna berhasil masuk dan sistem menampilkan halaman utama	Valid
VT-02	Kasus 2: Memasukkan *username* dan *password* yang tidak valid	Memasukkan *username* → memasukkan *password* → menekan tombol *login*	*Username* : 15062025 *Password* : 123456	Pengguna gagal masuk sistem	Muncul pesan invalid *username* atau *password*	Valid

4.5.1.2 *Validation testing* mengelola kamar

Tabel 4.35 menunjukkan pengujian validasi pada fungsional mengelola kamar. Terdapat tiga kasus pengujian pada modul mengelola kamar. Pertama menambah kamar. Kedua mengubah data kamar. Ketiga, mencari data kamar asrama.

Tabel 4.35 *Test Case* Mengelola Kamar

Modul	: Fungsional mengelola kamar
Deskripsi	: Menguji jalannya fungsi mengelola kamar yang berfungsi untuk menambah, mengubah dan mencari data kamar asrama
Traceability	: Kebutuhan Fungsional(KF-02, KF-03, KF-04), *Use Case Diagram* Mengelola Kamar, *Activity Diagram* Mengelola Kamar

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-03	Kasus 1: Menambahkan kamar	Membuka menu kelola kamar → menekan tombol tambah kamar → memilih gedung → mengisi nama kamar → mengisi kode kamar → menekan tombol simpan	Gedung: Al-azhar (kampus 1) Nama: Aisyah 1 Kode: D101	Data berhasil disimpan	Data berhasil disimpan	Valid
VT-04	Kasus 2: Mengubah data kamar	Membuka halaman kelola kamar → menekan tombol edit pada kamar yang dipilih →mengubah nama kamar → menekan tombol simpan	Kamar dipilih: D101 Nama pengganti: Aisyaah 2	Data berhasil diubah	Data berhasil diubah	Valid
VT-05	Kasus 3: Mencari data kamar berdasarkan kecocokan	Membuka halaman kelola kamar → menuliskan kata kunci	Kata pencarian: aisyah	Menampilkan kamar yang mengandung kaya aisyah	Menampilkan kamar yang mengandung kaya aisyah	Valid

	kata yang dicari	yang ingin dicari → melihat data yang dicari				

4.5.1.3 *Validation testing* mengelola kamar santri

Tabel 4.36 menunjukkan pengujian validasi pada fungsional mengelola kamar santri. Terdapat tiga kasus pengujian pada modul mengelola kamar santri. Pertama menambahkan santri yang belum memiliki kamar ke kamar. Kedua, menambahkan santri yang sudah memiliki kamar ke dalam kamar. Ketiga, menghapus santri dari data anggota kamar.

Tabel 4.36 *Test Case* Mengelola Kamar Santri

Modul	: Fungsional mengelola kamar santri
Deskripsi	: Menguji jalannya fungsi mengelola kamar santri yang berfungsi untuk menambah dan menghapus data santri kamar
Traceability	: Kebutuhan Fungsional(KF-05, KF-06, KF-10), *Use Case Diagram* Mengelola Kamar Santri

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-06	Kasus 1: Menambahkan santri yang belum memiliki kamar	Membuka halaman kelola kamar santri → menekan tombol tambah anggota → memilih kamar →memilih santri → menekan tombol simpan	Kamar: Aisyah 1 Santri: Andrea Nova Valerina	Santri berhasil ditambahkan ke kamar	Santri berhasil ditambahkan ke kamar	Valid
VT-07	Menambahkan santri yang sudah memiliki kamar	Membuka halaman kelola kamar santri → menekan tombol	Kamar: Aisyah 1 Santri: Andrea Nova Valerina	Santri gagal ditambahkan ke dalam kamar	Santri yang akan ditambah tidak ada didaftar pilihan	Valid

		tambah anggota → memilih kamar →memilih santri → menekan tombol simpan				
VT-08	Menghapus santri dari kamar	Membuka halaman kelola kamar santri → menekan tombol detail kamar pada kamar yang dipilih → menekan tombol hapus santri pada santri yang dipilih	Kamar: Aisyah 1 Santri: Andrea Nova Valerina	Santri berhasil dihapus dari kamar	Santri berhasil dihapus dari kamar	Valid

4.5.1.4 *Validation testing* mengelola murabbi kamar

Tabel 4.37 menunjukkan pengujian validasi pada fungsional mengelola murabbi kamar. Terdapat tiga kasus pengujian pada modul mengelola kamar santri. Pertama menambahkan murabbi. Kedua, mengubah data murabbi kamar. Ketiga, menghapus data murabbi kamar.

Tabel 4.37 *Test Case* Mengelola Murabbi Kamar

Modul	: Fungsional mengelola murabbi kamar
Deskripsi	: Menguji jalannya fungsi mengelola murabbi kamar yang berfungsi untuk menambah, mengubah dan menghapus data murabbi kamar
Traceability	: Kebutuhan Fungsional(KF-07, KF-08, KF-09), *Use Case Diagram* Mengelola Murabbi Kamar

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid

VT-09	Kasus 1: Menambah murabbi kamar	Menekan tombol tambah → semester akademik dan kampus →memilih murabbi masing-masing kamar → menekan tombol simpan	Kamar: • Aisyah 1 • Aisyah 2 • Aisyah 3 Murabbi: • Dzakiyatul Afifah • Nadia Saphira Cahyani, S.Ag • Nadia Saphira Cahyani, S.Ag	Murabbi Berhasil Ditambahkan	Murabbi Berhasil Ditambahkan	Valid
VT-10	Kasus 2: Mengubah murabbi kamar	Memilih kamar yang akan diubah → menekan tombol edit →memilih murabbi pengganti → menekan tombol simpan	Kamar: Aisyah 1 Murabbi pengganti: Selda Monazir, S.Pd	Murabbi berhasil diubah	Murabbi berhasil diubah	Valid
VT-11	Kasus 3: Menghapus murabbi kamar	Memilih kamar yang akan dihapus murabbinya → menekan tombol hapus	Kamar: Aisyah 3	Murabbi kamar berhasil dihapus	Murabbi kamar berhasil dihapus	Valid

4.5.1.5 *Validation testing* mengelola item kemandirian

Tabel 4.38 menunjukkan pengujian validasi pada fungsional mengelola item kemandirian. Terdapat tiga kasus pengujian pada modul mengelola item kemandirian. Pertama menambahkan item. Kedua, mengubah data item. Ketiga, menghapus data item kemandirian.

Tabel 4.38 *Test Case* Mengelola Item Kemandirian

Modul	: Fungsional mengelola item kemandirian

Deskripsi	: Menguji jalannya fungsi mengelola item kemandirian yang berfungsi untuk menambah, mengubah dan menghapus data item kemandirian
Traceability	: Kebutuhan Fungsional(KF-12, KF-13, KF-14,KF-15), *Use Case Diagram* Mengelola Item Kemandirian

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-12	Kasus 1: Menambah item kemandirian	Membuka menu kelola item kemandirian → menekan tombol tambah → memasukkan data item → menekan tombol simpan	Nama aktivitas: memotong kuku Alias: kuku Hari: Jumat	Item berhasil ditambahkan	Item berhasil ditambahkan	Valid
VT-13	Kasus 2: Mengubah item kemandirian	Membuka menu kelola item kemandirian→ menekan tombol edit pada item yang dipilih → mengganti data item → menekan tombol simpan	Nama Aktivitas: membersihkan kasur Alias: kasur Hari pengganti : senin, rabu, jumat, minggu	Item berhasil diubah	Item berhasil diubah	Valid
VT-14	Kasus 3: Menghapus item kemandirian	Membuka halaman kelola item kemandirian → menekan tombol hapus pada item yang dipilih	Nama aktivitas: merapikan loker	Item berhasil dihapus	Item berhasil dihapus	Valid

4.5.1.6 *Validation testing* mengelola item peribadatan

Tabel 4.39 menunjukkan pengujian validasi pada fungsional mengelola item peribadatan. Terdapat tiga kasus pengujian pada modul mengelola item peribadatan. Pertama menambahkan item. Kedua, mengubah data item. Ketiga, menghapus data item kemandirian.

Tabel 4.39 *Test Case* Mengelola Item Peribadatan

Modul	: Fungsional mengelola item peribadatan
Deskripsi	: Menguji jalannya fungsi mengelola item peribadatan yang berfungsi untuk menambah, mengubah dan menghapus data item peribadatan
Traceability	: Kebutuhan Fungsional(KF-23, KF-24, KF-25,KF-26), *Use Case Diagram* Mengelola Item Peribadatan

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-15	Kasus 1: Menambah item peribadatan	Membuka menu kelola item peribadatan → menekan tombol tambah → memasukkan data item → menekan tombol simpan	Kategori: Sholat wajib Nama aktivitas: Sholat subuh Alias: subuh	Item berhasil ditambahkan	Item berhasil ditambahkan	Valid
VT-16	Kasus 2: Mengubah item peribadatan	Membuka menu kelola item peribadatan→ menekan tombol edit pada item yang dipilih → mengganti data item → menekan tombol simpan	Nama Aktivitas: Sholat malam Nama pengganti: Sholat tahajud	Item berhasil diubah	Item berhasil diubah	Valid

VT-17	Kasus 3: Menghapus item peribadatan	Membuka halaman kelola item peribadatan → menekan tombol hapus pada item yang dipilih	Nama aktivitas: sholat dhuha	Item berhasil dihapus	Item berhasil dihapus	Valid

4.5.1.7 *Validation testing* mengelola jurnal kemandirian

Tabel 4.40 menunjukkan pengujian validasi pada fungsional mengelola jurnal kemandirian. Terdapat tiga kasus pengujian pada modul mengelola jurnal kemandirian. Pertama menambahkan jurnal. Kedua, mengubah jurnal. Ketiga, menghapus jurnal.

Tabel 4.40 *Test Case* Mengelola Jurnal Kemandirian

Modul	: Fungsional mengelola jurnal kemandirian
Deskripsi	: Menguji jalannya fungsi mengelola jurnal kemandirian yang berfungsi untuk menambah, mengubah dan menghapus jurnal kemandirian
Traceability	: Kebutuhan Fungsional(KF-16, KF-17, KF-18,KF-19), *Use Case Diagram* Mengelola Jurnal Kemandirian

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-18	Kasus 1: Menambah jurnal	Membuka menu kelola jurnal kemandirian → menekan tombol tambah → memilih kamar dan tanggal → memasukkan data kehadiran → menekan tombol simpan	Kamar : Aisyah 1 Tanggal: 20 April 2021 Kehadiran: Semua santri melaksanakan kemandirian	Jurnal berhasil ditambahkan	Jurnal berhasil ditambahkan	Valid

VT-19	Kasus 2: Mengubah data kehadiran jurnal	Membuka menu kelola jurnal kemandirian → menekan tombol edit pada item yang dipilih → mengubah data kehadiran santri → menekan tombol simpan	Kamar : Aisyah 1 Tanggal: 20 April 2021 Kehadiran: mengganti salah satu anak menjadi tidak hadir	Jurnal berhasil diubah	Jurnal berhasil diubah	Valid
VT-20	Kasus 3: Menghapus jurnal	Membuka menu kelola jurnal kemandirian → menekan tombol hapus pada item yang dipilih	Kamar : Aisyah 1 Tanggal: 20 April 2021	Jurnal berhasil dihapus	Jurnal berhasil dihapus	Valid

4.5.1.8 *Validation testing* mengelola jurnal peribadatan

Tabel 4.41 menunjukkan pengujian validasi pada fungsional mengelola jurnal peribadatan. Terdapat tiga kasus pengujian pada modul mengelola jurnal peribadatan. Pertama menambahkan jurnal. Kedua, mengubah jurnal. Ketiga, menghapus jurnal.

Tabel 4.41 *Test Case* Mengelola Jurnal Peribadatan

Modul	: Fungsional mengelola jurnal peribadatan
Deskripsi	: Menguji jalannya fungsi mengelola jurnal peribadatan yang berfungsi untuk menambah, mengubah dan menghapus jurnal peribadatan
Traceability	: Kebutuhan Fungsional(KF-27, KF-28, KF-29,KF-30), *Use Case Diagram* Mengelola Jurnal Peribadatan

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid

VT-21	Kasus 1: Menambah jurnal	Membuka menu kelola jurnal peribadatan → menekan tombol tambah → memilih kamar dan tanggal → memasukkan data kehadiran → menekan tombol simpan	Kamar : Aisyah 1 Tanggal: 20 April 2021 Kehadiran: Semua santri hadir	jurnal berhasil ditambahkan	Jurnal berhasil ditambahkan	Valid
VT-22	Kasus 2: Mengubah data kehadiran jurnal	Membuka menu kelola jurnal peribadatan → menekan tombol edit pada item yang dipilih → mengubah data kehadiran santri → menekan tombol simpan	Kamar : Aisyah 1 Tanggal: 20 April 2021 Kehadiran: mengganti salah satu anak menjadi tidak hadir	Jurnal berhasil diubah	Jurnal berhasil diubah	Valid
VT-23	Kasus 3: Menghapus jurnal	Membuka menu kelola jurnal peribadatan → menekan tombol hapus pada item yang dipilih	Kamar : Aisyah 1 Tanggal: 20 April 2021	Jurnal berhasil dihapus	Jurnal berhasil dihapus	Valid

4.5.1.9 *Validation testing* mengelola nilai ujian kemandirian

Tabel 4.42 menunjukkan pengujian validasi pada fungsional mengelola nilai ujian kemandirian. Pengujian pada modul mengelola nilai ujian kemandirian dilakukan untuk mengetahui validitas proses memasukkan dan mengubah nilai ujian.

Tabel 4.42 *Test Case* Mengelola Niai Ujian Kemandirian

Modul	: Fungsional mengelola nilai ujian kemandirian
Deskripsi	: Menguji jalannya fungsi mengelola nilai ujian kemandirian yang berfungsi untuk menambah dan mengubah nilai ujian kemandirian
Traceability	: Kebutuhan Fungsional(KF-20, KF-21), *Use Case Diagram* Mengelola Nilai Ujian Kemandirian

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-24	Kasus 1: Menambah nilai ujian	Membuka menu kelola nilai ujian kemandirian → menekan tombol detail pada kamar yang dipilih → memilih level ujian → memasukkan nilai ujian → menekan tombol simpan	Kamar : Aisyah 1 Level: SMA Nilai: 90	Nilai berhasil ditambahkan	Nilai berhasil ditambahkan	Valid
VT-25	Kasus 2: Mengubah nilai ujian	Membuka menu kelola nilai ujian kemandirian → menekan tombol detail pada kamar yang sudah pernah dinilai→ memilih	Kamar : Aisyah 1 Level: SMA Nilai: 85	Nilai berhasil disimpan	Nilai berhasil disimpan	Valid

		level ujian → memasukkan nilai ujian → menekan tombol simpan				

4.5.1.10 *Validation testing* mengelola nilai ujian peribadatan

Tabel 4.43 menunjukkan pengujian validasi pada fungsional mengelola nilai ujian peribadatan. Pengujian pada modul mengelola nilai ujian peribadatan dilakukan untuk mengetahui validitas proses memasukkan dan mengubah nilai ujian.

Tabel 4.43 *Test Case* Mengelola Niai Ujian Peribadatan

Modul		: Fungsional mengelola nilai ujian peribadatan				
Deskripsi		: Menguji jalannya fungsi mengelola nilai ujian peribadatan yang berfungsi untuk menambah dan mengubah nilai ujian peribadatan				
Traceability		: Kebutuhan Fungsional(KF-31, KF-32), *Use Case Diagram* Mengelola Nilai Ujian Peribadatan				
Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-26	Kasus 1: Menambah nilai ujian	Membuka menu kelola nilai ujian peribadatan → menekan tombol detail pada kamar yang dipilih → memasukkan nilai ujian → menekan tombol simpan	Kamar : Aisyah 1 Nilai: 90	Nilai berhasil ditambahkan	Nilai berhasil ditambahkan	Valid
VT-27	Kasus 2: Mengubah nilai ujian	Membuka menu kelola nilai ujian peribadatan → menekan	Kamar : Aisyah 1 Nilai: 85	Nilai berhasil disimpan	Nilai berhasil disimpan	Valid

		tombol detail pada kamar yang sudah pernah dinilai→ memasukkan nilai ujian → menekan tombol simpan				

4.5.1.11 *Validation testing* melihat ledger kemandirian

Tabel 4.44 menunjukkan pengujian validasi pada fungsional melihat ledger kemandirian. Pengujian ini dilakukan untuk mengetahui validitas modul ledger kemandirian.

Tabel 4.44 *Test Case* Melihat Ledger Kemandirian

Modul	: Fungsional melihat ledger kemandirian
Deskripsi	: Menguji jalannya fungsi melihat ledger kemandirian yang berfungsi untuk melihat laporan nilai kuantitas dan kualitas program kemandirian santri
Traceability	: Kebutuhan Fungsional(KF-22), *Use Case Diagram* melihat ledger kemandirian

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-28	Kasus 1: Melihat ledger	Membuka menu ledger kemandirian → menekan tombol detail pada kamar yang dipilih	Kamar : Aisyah 1	Menampilkan nilai kualitas dan kuantitas kemandirian	Menampilkan nilai kualitas dan kuantitas kemandirian	Valid

4.5.1.12 *Validation testing* melihat ledger peribadatan

Tabel 4.45 menunjukkan pengujian validasi pada fungsional melihat ledger peribadatan. Pengujian ini dilakukan untuk mengetahui validitas modul ledger peribadatan.

Tabel 4.45 *Test Case* Melihat Ledger Peribadatan

Modul	: Fungsional melihat ledger peribadatan
Deskripsi	: Menguji jalannya fungsi melihat ledger peribadatan yang berfungsi untuk melihat laporan nilai kuantitas dan kualitas program peribadatan santri
Traceability	: Kebutuhan Fungsional(KF-22), *Use Case Diagram* melihat ledger peribadatan

Kode	Kasus uji	Langkah pengujian	Data uji	Ekspektasi	Hasil	Valid/ tidak valid
VT-29	Kasus 1: Melihat ledger	Membuka menu ledger peribadatan → menekan tombol detail pada kamar yang dipilih	Kamar : Aisyah 1	Menampilkan nilai kualitas dan kuantitas peribadatan	Menampilkan nilai kualitas dan kuantitas peribadatan	Valid

4.6 *User Acceptence Testing*

Pengujian *user acceptance testing* (*UAT*) bertujuan untuk mengetahui seberapa tingkat penerimaan pengguna terhadap sistem yang telah dikembangkan, sehingga memberikan gambaran yang lebih baik kepada pengembang akan terpenuhinya kebutuhan pengguna dengan adanya sistem. Pengujian *UAT* dilakukan kepada dua pengguna utama sistem yaitu kepala kepesantrenan dan murabbi sebanyak empat responden. Pengujian dilakukan dengan meminta pengguna sistem tersebut untuk melakukan beberapa tugas. Hal ini dilakukan untuk memberikan gambaran mengenai sistem yang akan diuji.

Setelah serangkaian tugas telah dilakukan, langkah selanjutnya ialah responden diminta untuk memberikan penilaian terhadap beberapa pernyataan yang diberikan. Pernyataan yang diberikan tersebut berdasarkan atas empat kriteria pengujian pada *UAT*. Keempat kriteria tersebut antara lain *Functional Correctness and Completeness, Usability, Timeliness,* dan *Realiability and Availability*. Pernyataan dalam melakukan *UAT* dapat dilihat pada Tabel 4.46.

Tabel 4.46 Pernyataan *User Acceptance Testing*

No	Kriteria UAT	Definisi	Pernyataan
1	*Functional Correctness and Completeness*	Menggambarkan bagaimana sistem yang telah dibangun telah	1. Sistem dapat memberikan respons yang baik dan jelas terhadap masukan yang dimasukkan

No	Kriteria UAT	Definisi	Pernyataan
		memenuhi kebutuhan pengguna	
2	*Usability*	Menunjukkan seberapa mudah sistem yang telah dibangun untuk digunakan, dipahami, dan dipelajari pengguna	1. Alur kerja sistem sederhana dan mudah digunakan 2. Fitur-fitur pada sistem mudah dipahami 3. Informasi yang ditampilkan sistem mudah dipahami
3	*Timeliness*	Menggambarkan seberapa cepat sistem digunakan untuk menghasilkan informasi yang dibutuhkan pengguna	1. Sistem cepat dalam memberikan informasi yang diinginkan 2. Sistem mempercepat proses pelaporan nilai kemandirian dan peribadatan santri
4	*Realiability and Availability*	Menunjukkan apakah layanan dari sistem yang telah dibangun mampu diakses dengan mudah	1. Sistem membantu dalam proses penilaian kegiatan kemandirian dan peribadatan kapanpun dan dimanapun

Setelah kriteria dan pernyataan *UAT* didefinisikan, kemudian kriteria pernyataan dibagikan kepada responden menggunakan kuesioner berupa skala *Likert*. Responden diminta memilih salah satu jawaban yang telah disediakan. Pilihan jawaban ada 5 pilihan mulai dari sangat setuju hingga sangat tidak setuju. Data kualitatif diubah berdasarkan bobot skor seperti pada Tabel 4.47.

Tabel 4.47 Bobot Skala *Likert*

No	Kategori	Skor
1	Sangat Setuju	5
2	Setuju	4
3	Netral	3
2	Tidak setuju	2
1	Sangat Tidak setuju	1

Sumber: Diadaptasi dari Sugiono (2013)

Jawaban responden kemudian dihitung persentase penerimaan menggunakan Persamaan 4.1. Persentase penerimaan pengguna didapatkan dari total skor jawaban responden dibagi total skor maksimal yang bisa didapat dikali seratus persen.

$$P = \frac{\Sigma \mathbf{x}}{\Sigma x_i} \times 100\% \quad (4.1)$$

Keterangan:

P = Persentase penerimaan

$\Sigma \mathbf{x}$ = Total skor responden

Σx_i = Total skor maksimum

Kuesioner didapatkan dari empat responden yang terlibat dalam pengujian. Hasil analisis jawaban dari pernyataan *UAT* yang sudah diisi oleh responden dapat dilihat pada Tabel 4.48.

Tabel 4.48 Hasil *User Acceptance Testing*

No	Pernyataan	Penilaian					Σx	Σx_i	P
		1	2	3	4	5			
Functional Correctness and Completeness									
1	Sistem dapat memberikan respons yang baik dan jelas terhadap masukan yang dimasukkan				4		16	20	80%
Rata-rata									**80%**
Usability									
1	Alur kerja sistem sederhana dan mudah digunakan				4		16	20	80%
2	Fitur-fitur pada sistem mudah dipahami				4		16	20	80%
3	Informasi yang ditampilkan sistem mudah dipahami				3	1	17	20	85%
Rata-rata									**81,7%**
Timeliness									
1	Sistem cepat dalam memberikan informasi yang diinginkan				2	2	18	20	90%
2	Sistem mempercepat proses pelaporan nilai kemandirian dan peribadatan santri				3	1	17	20	85%
Rata-rata									**87,5%**
Realiability and Availability									
1	Sistem membantu dalam proses penilaian kegiatan kemandirian dan peribadatan kapanpun dan dimanapun				3	1	17	20	85%
Rata-rata									**85%**

Hasil perhitungan pada persentase penerimaan kemudian dapat disesuaikan dengan skala kategori penerimaan pada Tabel 4.49 untuk mengetahui tingkat penerimaan dari masing-masing kriteria.

Tabel 4.49 Skala Kategori Penerimaan

Interval	Skala
0% - 19.99%	Sangat Tidak Setuju
20% - 39.99%	Tidak Setuju
40% - 59.99%	Netral
60% - 79.99%	Setuju
80% - 100%	Sangat Setuju

Dari analisis hasil perhitungan kriteria pada *UAT* didapatkan hasil bahwa pengguna sangat setuju bahwa sistem yang dikembangkan sesuai dengan apa yang dibutuhkan dengan skor sebesar 80%. Pengguna juga sangat setuju bahwa sistem mudah digunakan dengan skor 81,7%. Selain itu pengguna sangat setuju bahwa sistem mempercepat proses penilaian dan pelaporan kemandirian dan peribadatan sebesar 87,5%. Terakhir, pengguna juga sangat setuju sistem membantu proses kegiatan kemandirian dan peribadatan kapanpun dan dimanapun sebesar 85%.

BAB 5 PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang sudah dilakukan dapat diambil beberapa kesimpulan dari pengembangan sistem informasi manajemen kepengasuhan di pesantren sebagai berikut:

1. Berdasarkan pengujian penerimaan pengguna, sistem informasi dapat mempercepat aktivitas pencatatan nilai santri. Hal ini dapat dilihat dari hasil pengujian penerimaan pengguna bahwa sangat setuju sistem yang telah dibangun mudah untuk digunakan, dipahami, dan dipelajari pengguna. Selain itu pengguna juga sangat setuju sistem membantu dalam proses penilaian kegiatan kemandirian dan peribadatan kapanpun dan dimanapun.
2. Sistem informasi dapat mempercepat pencarian laporan nilai santri. Berdasarkan pengujian penerimaan pengguna, pengguna sangat setuju sistem mempercepat proses pelaporan nilai kemandirian dan peribadatan santri. Selain itu pengguna juga sangat setuju sistem yang digunakan mempercepat menghasilkan informasi yang dibutuhkan pengguna.
3. Sistem informasi yang dikembangkan dapat dikatakan efektif digunakan dalam proses penilaian kepengasuhan santri karena menghasilkan informasi yang dapat diterima dan mampu memenuhi harapan informasi secara tepat waktu, akurat, dan dapat dipercaya.

5.2 Saran

Saran yang dapat diberikan sebagai bahan pertimbangan pada pengembangan lanjut dari sistem informasi manajemen kepengasuhan santri. Pada pengembangan lebih lanjut sistem dapat dikembangkan berbasis aplikasi perangkat bergerak agar meningkatkan mobilitas pengasuh dalam mengoperasikan sistem. Selain itu dapat dikembangkan sistem informasi yang dapat melengkapi aktivitas kepesantrenan lainnya seperti, tahfidz, *checklist* kebersihan kamar dan ekstrakurikuler santri.

DAFTAR PUSTAKA

Anwar, A., 2014. A Review of RUP (Rational Unified Process). *International Journal of Software Engineering (IJSE)*, 5(2), 8-24.

Booch, G., Rumbaugh, J., & Jacobson, I., 2005. *Unified Modeling Language User Guide, The, Second Edition*. Addison Wesley Professional.

Erliyah Nurul Jannah, el al., 2018. SISTEM INFORMASI MANAJEMEN KEGIATAN SANTRI PONDOK PESANTREN BERBASIS WEB. *Seminar Nasional Teknologi Informasi dan Komunikasi 2018 (SENTIKA 2018),* 23-24 Maret 2018, pp. 260-267.

Gaol, L. Jimmy., 2008. *Sistem Informasi Manajemen Pemahaman dan Aplikasi*. Jakarta: Penerbit PT Grasindo.

George, J. F., Batra, D., Valacich, J. S., & Hoffer, J. A., 2004. *Object-Oriented Systems Analysis and Design*. Upper Saddle River, NJ: Prentice Hall.

Harrington, H. J., 1991. *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness*. New York: McGrawHill.

Hasibuan, Zainal A., 2007. *Metodologi Penelitian pada Bidang Ilmu Komputer dan Teknologi Informasi.* Depok: Universitas Indonesia.

Hendri Murti Susanto, et al., 2015. Pengembangan Sistem Informasi Manajemen Pendidik Dan Tenaga Kependidikan. *Jurnal Pendidikan Humaniora*, vol. 3, Juni 2015, 93–105., journal.um.ac.id/index.php/jph.

IEEE Computer Society, 2004. *SWEBOK V3.0: Guide to the Software Engineering Body of Knowledge.* [pdf] The Institute of Electrical and Electronics Engineers, Inc. Tersedia di: <www.swebok.org,.https://www.computer.org/education/b,> [diaskes 19 Oktober 2020].

Indrajit, 2001. *Analisis dan Perancangan Sistem Berorientasi Object*. Bandung: Informatika.

Jogiyanto, H.M., 2005, *Analisa dan Desain Sistem Informasi: Pendekatan Terstruktur Teori dan Praktik Aplikasi Bisnis*. Yogyakarta: Andi.

Kadir, A., 2003. *Pengenalan Sistem Informasi*. Yogyakarta: Andi.

Kruchten, P., 2000. *The Rational Unified Process. Reading*, MA: Addison-Wesley.

Maksudin., 2006. *Pendidikan Nilai Sistem Boarding School di SMP IT Abu Bakar Hasil Penelitian Untuk Disertasi* . Yogyakarta: Program Pasca Sarjana UIN Sunan Kalijaga. Marakas dan O'Brien, 2010. *Management System Information*. New York: McGraw Hill.

Nizar. S., 2007. *Sejarah Pendidikan Islam : Menelusuri Jejak Sejarah Pendidikan Era Rasulullah Sampai Indonesia*. Jakarta: Kencana Prenada Media Group.

Object Management Group. 2011. *Business Process Model and Notation (BPMN).version 2.0,* [pdf], (https://www.omg.org/spec/BPMN/2.0/PDF, diakses 17 April 2020).

Pressman, R. S., 2010. *Software Engineering A Practiotioner's Aproach. Seventh Edition*. New York: The McGraw-Hill Companies, Inc.

Sommerville, Ian., 2011. *Software Engineering: Ninth Edition*. New York, Addison-Wesley.

Sugiyono., 2013. *Metode Penelitian Pendidikan Pendekatan Kuantitatif, Kualitatif, dan R&D*. Bandung: Alfabeta.

Supono, dan Virdiandry Putratama., 2016. *Pemograman Web Dengan Menggunakan PHP dan Framework Codeigniter*. Yogyakarta: Deepublish (Grup Penerbitan CV Budi Utama).

Sutabri, T., 2012. *Konsep sistem informasi*. Yogyakarta: Andi.

Visual-paradigm.com. *How To Model MVC Framework With UML Sequence Diagram?*. [online] tersedia di: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/how-to-model-mvc-with-uml-sequence-diagram/> [diakses pada 18 Desember 2020].

Wahono, Romi,. 2003. Analyzing Requirements Engineering Problems. *Proceedings of the IECI Japan Workshop,* 55-58.

Weske, M., 2012. *Business Process Management Concepts Languages, Architectures*. New York: Springer.

Yakub, 2012. *Pengantar Sistem informasi*. Yogyakarta: Graha Ilmu.

LAMPIRAN A RANCANGAN WAWANCARA

Narasumber :

Jabatan :

Lokasi :

No.	Pertanyaan	Jawaban
1	Apa saja kegiatan kepesantrenan yang terdapat di Tazkia IIBS?	
2	Bagaimana proses kegiatan kepengasuhan berlangsung?	
3	Bagaimana pembuatan laporan nilai hasil kepengasuhan?	
4	Siapa saja yang berperan dalam proses pembuatan nilai kepengasuhan?	
5	Apa dan bagaimana kendala yang dialami dalam penilaian kepengasuhan?	
6	Bagaimana bentuk pemanfaatan sistem informasi dalam proses kepengasuhan?	
7	Hal apa yang ingin ditingkatkan dengan adanya sistem informasi manajemen kepengasuhan di Tazkia IIBS?	

LAMPIRAN B HASIL WAWANCARA

Narasumber : Nanang Setyobudi, S.Fil.I

Jabatan : Kepala Kepesantrenan Tazkia IIBS

Lokasi : Tazkia IIBS

No.	Pertanyaan	Jawaban
1	Apa saja kegiatan kepesantrenan yang terdapat di Tazkia IIBS?	Kegiatan kepesantrenan yang ada di Tazkia cukup banyak. Mulai dari aktivitas yang bersifat harian, mingguan, bulanan maupun semester. Dalam kegiatan harian contohnya aktivitas kepengasuhan yaitu peribadatan dan kemandirian, selain itu ada tahfidz Quran. Kegiatan tersebut dilakukan secara rutin setiap hari oleh para santri. Ada juga aktivitas mingguan yaitu enrichment/pengembangan diri melalui kegiatan-kegiatan seperti memanah, berkuda, berenang dan lainnya. Untuk kegiatan bulanan terdapat ujian tahfidz bulanan. Sedangkan untuk kegiatan semester terapat ujian praktik kemandirian dan ibadah yang biasanya dilakukan pada akhir semester.
2	Bagaimana proses kegiatan kepengasuhan berlangsung?	Terdapat dua proses utama yang ada pada kegiatan kepengasuhan yaitu kemandirian dan peribadatan. Kemandirian adalah penilaian terhadap kualitas dan kuantitas kemandirian yang dilakukan santri selama mondok di Tazkia IIBS. Sedangkan peribadatan adalah penilaian terhadap kualitas dan kuantitas aktivitas ibadah santri. Penilaian tersebut dilakukan oleh pengasuh atau jika di Tazkia dikenal dengan sebutan murabbi/murabbiah. Proses tersebut dimulai ketika awal semester kepala kepesantrenan melakukan pembagian santri ke kamar-kamar yang sudah ditentukan. Setiap kamar berisi sekitar delapan santri. Selain itu kepala kepesantrenan juga membagi murabbi untuk setiap kamar. Biasanya setiap murabbi bertanggung jawab sekitar tiga kamar atau dengan rasio 1:24. Setiap murabbi tersebut akan menjadi pengasuh santri-santri kamar yang dimilikinya selama satu semester. Seorang murabbi dapat bertindak sebagai pemimpin, guru, orang tua, dan sahabat bagi santrinya. Dalam proses kepengasuhan murabbi juga bertugas melakukan penilaian terhadap ibadah dan kemandirian santri. Setiap hari murabbi bertugas mencatat keterlaksanaan aktivitas santri pada sebuah jurnal yang sudah dibuat oleh kepala kepesantrenan sebelumnya. Jurnal tersebut berisi aktivitas-aktivitas

		yang dinilai dalam proses kemandirian maupun peribadatan. Beberapa contoh aktivitas kemandirian diantaranya menata baju, memotong kuku, belajar malam dan lainnya. Sedangkan untuk peribadatan misalnya semua solat fardhu, solat tahajud dan ibadah lainnya. Penilaian tersebut digunakan sebagai nilai kuantitas santri. Selanjutnya ada nilai ujian yang dilakukan sekali sebagai nilai kualitas. Semua nilai dari jurnal harian dan ujian di rekapitulasi oleh murabbi, kemudian dimasukkan ke ledger penilaian. Ledger terebut dilaporkan ke kepala kepesantrenan.
3	Bagaimana pembuatan ledger nilai hasil kepengasuhan?	Proses pembuatan laporan atau ledger penilaian kemandirian dan peribadatan sama. Pertama hasil dari jurnal harian direkapitulasi jumlah tidak keterlaksanaan setiap santri. Hasil rekapitulasi tersebut menjadi nilai kualitatif. Nilai kualitatif memiliki syarat maksimal 6 kali ketidakterlaksanaan agar dinyatakan tuntas. Kemudian ada nilai kualitatif. Yaitu nilai yang berasal dari hasil ujian praktik. Nilai kualitatif mempunyai syarat rata-rata minimal adalah 85 agar dinyatakan tuntas. Antara nilai kualitas dan kuantitas memiliki bobot yang sama. Agar santri dapat dinyatakan tuntas dari proses kemandirian atau peribadatan maka nilai kualitas dan nilai kuantitas keduanya harus tuntas. Apabila salah satu tidak tuntas, maka santri dinyatakan tidak tuntas.
4	Siapa saja yang berperan dalam proses pembuatan nilai kepengasuhan?	Ada kepala kepesantrenan dan murabbi/murabbiah
5	Apa dan bagaimana kendala yang dialami dalam penilaian kepengasuhan?	Salah satu kendala yang ada adalah murabbi harus mencatat jurnal harian dalam lembaran-lembaran kertas setiap hari. Hal tersebut tentu saja sangat tidak efisien karena menghabiskan banyak kertas. Apalagi ketika melakukan rekapitulasi secara manual. Murabbi sangat diberatkan karena harus menghitung jumlah secara manual dari jurnal tersebut. Belum lagi risiko kehilangan atau kerusakan jurnal tersebut yang mungkin terkena air atau lainnya. Hal tersebut sangat memberatkan murabbi.
6	Bagaimana bentuk pemanfaatan sistem informasi dalam proses kepengasuhan?	Saat ini belum ada penerapan sistem informasi dalam proses kepengasuhan. Namun dari lembaga sendiri sangat mengharapkan adanya sistem yang mampu memudahkan proses penilaian kepengasuhan yang ada. Keinginan tersebut dibuktikan dengan adanya rencana strategis

		pembuatan sistem informasi yang mendukung seluruh kegiatan edukasi di Tazkia IIBS.
7	Hal apa yang ingin ditingkatkan dengan adanya sistem informasi manajemen kepengasuhan di Tazkia IIBS?	Hal ingin ditingkatkan dengan adanya sistem informasi manajemen adalah kemudahan proses penyaluran informasi baik dari kepala kepesantrenan ke murabbi atau sebaliknya. Seperti proses pembagian kamar dan pelaporan hasil nilai santri. Selain itu kami berharap dengan adanya sistem informasi dapat meringankan beban murabbi dalam melakukan penilaian. Sehingga waktunya tidak habis terkuras dengan dokumen-dokumen yang ada.

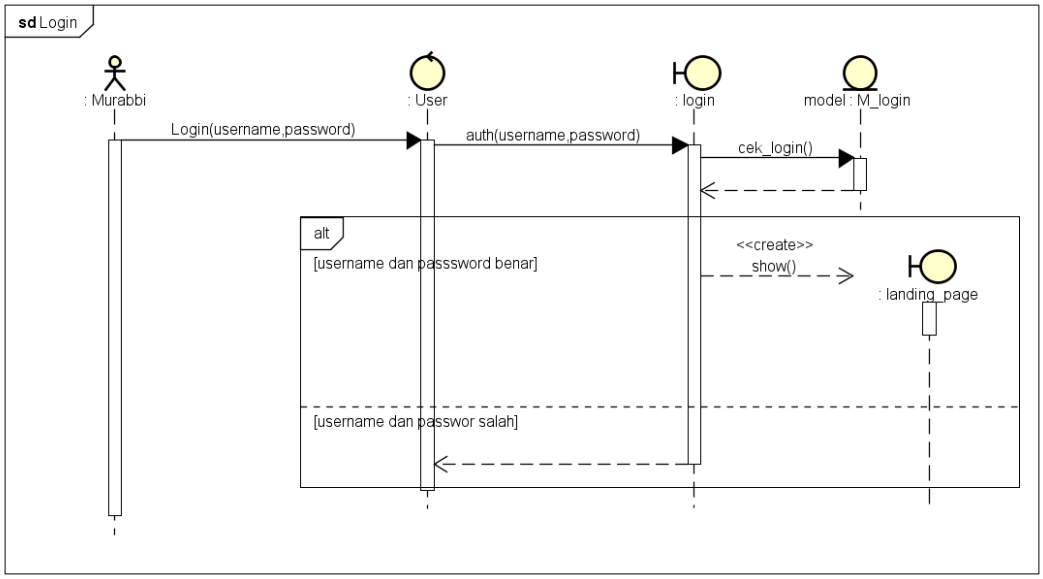
Malang, 2020

Kepala Kepesantrenan Tazkia IIBS

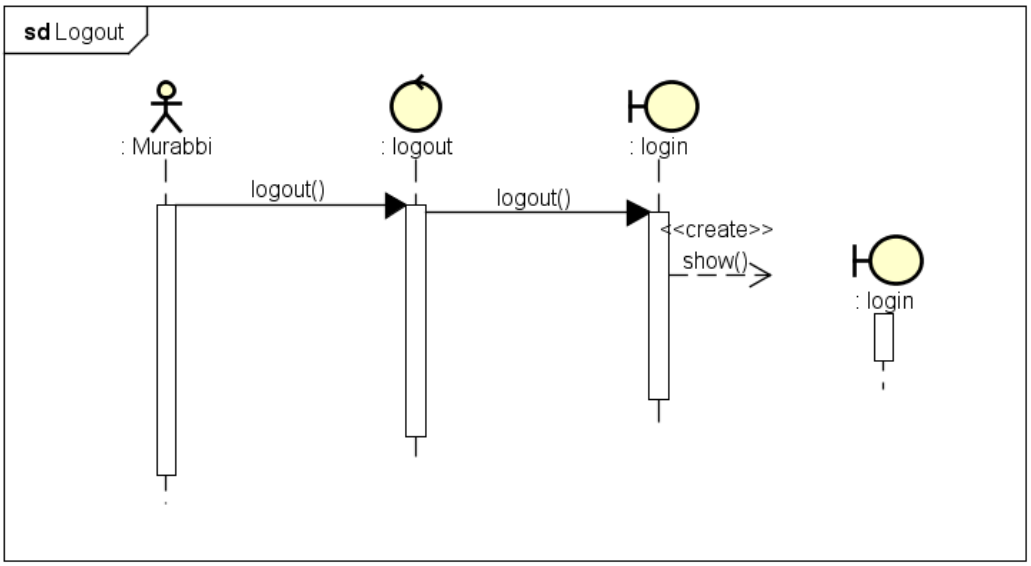
Nanang Setyobudi, S.Fil.I

LAMPIRAN C *SEQUENCE DIAGRAM*

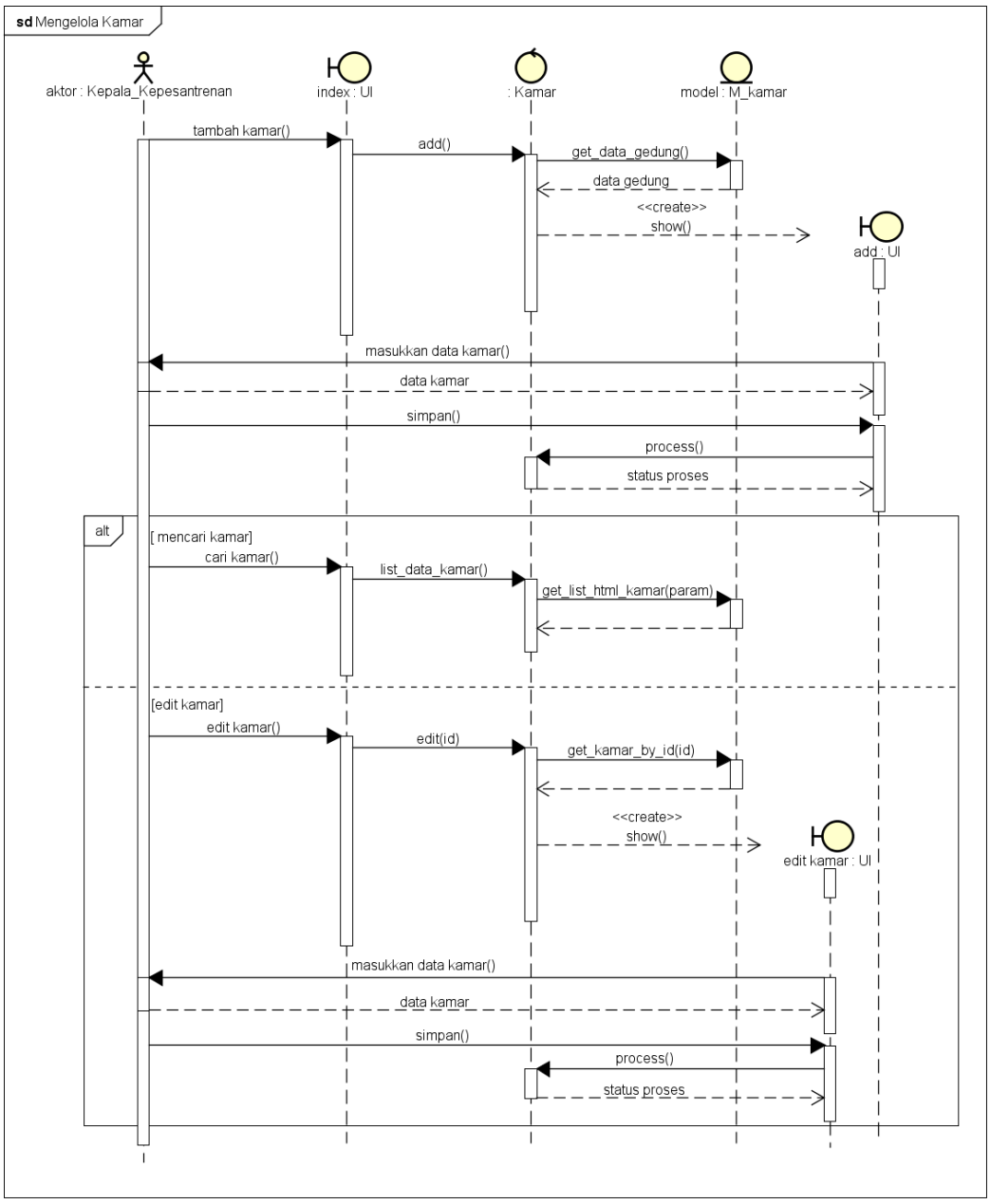
1. *Sequence Diagram Login*



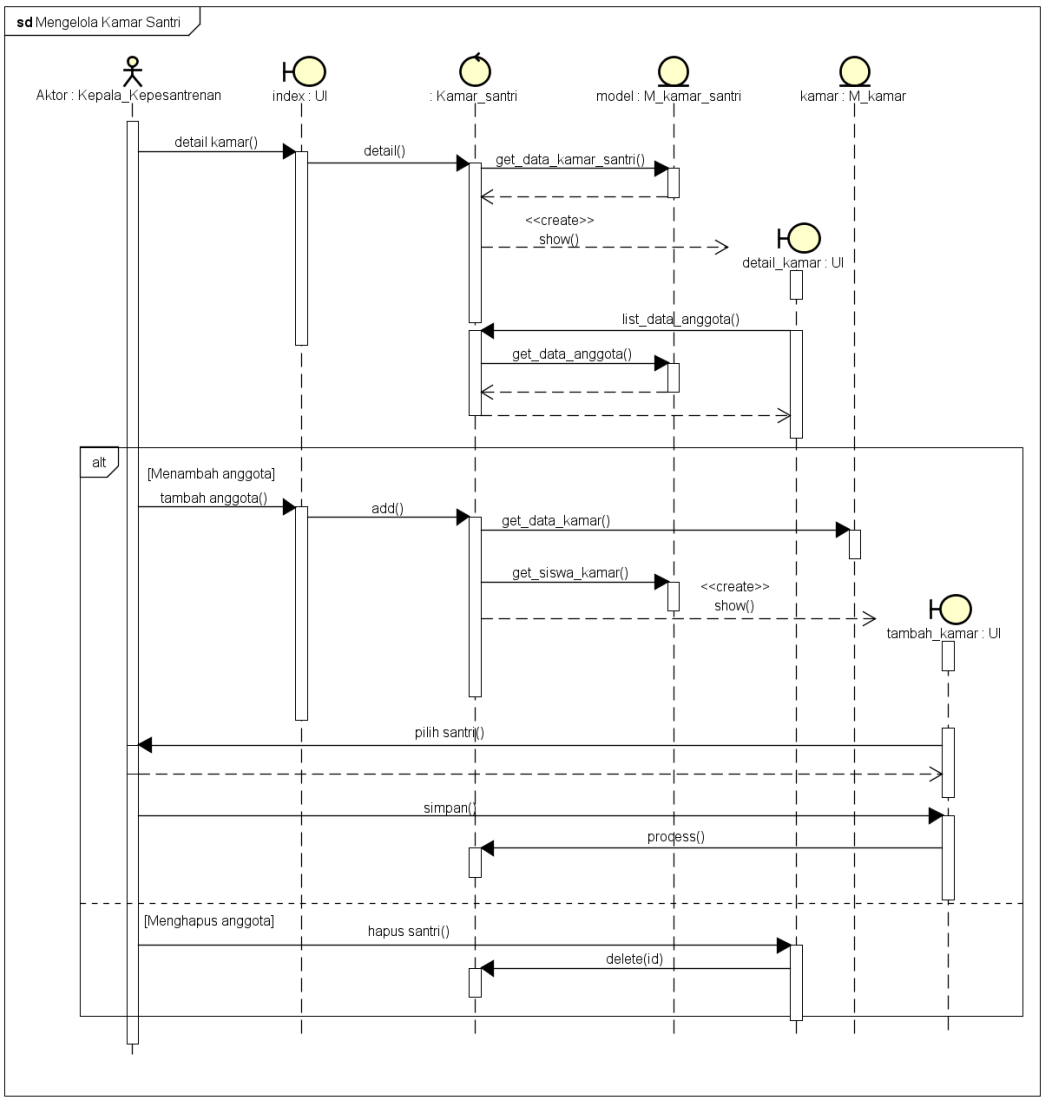
2. *Sequence Diagram Logout*



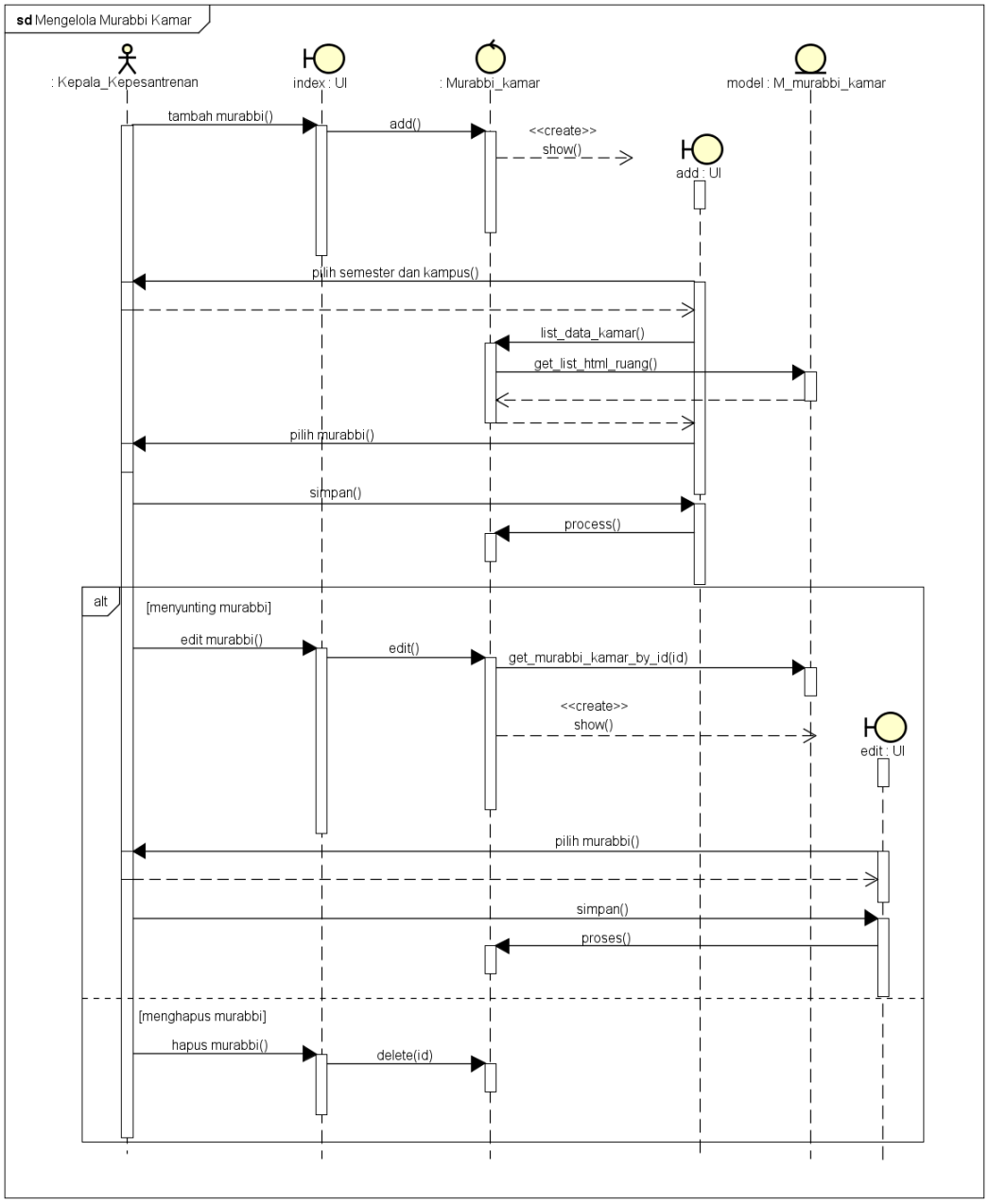
3. *Sequence Diagram* Mengelola Kamar



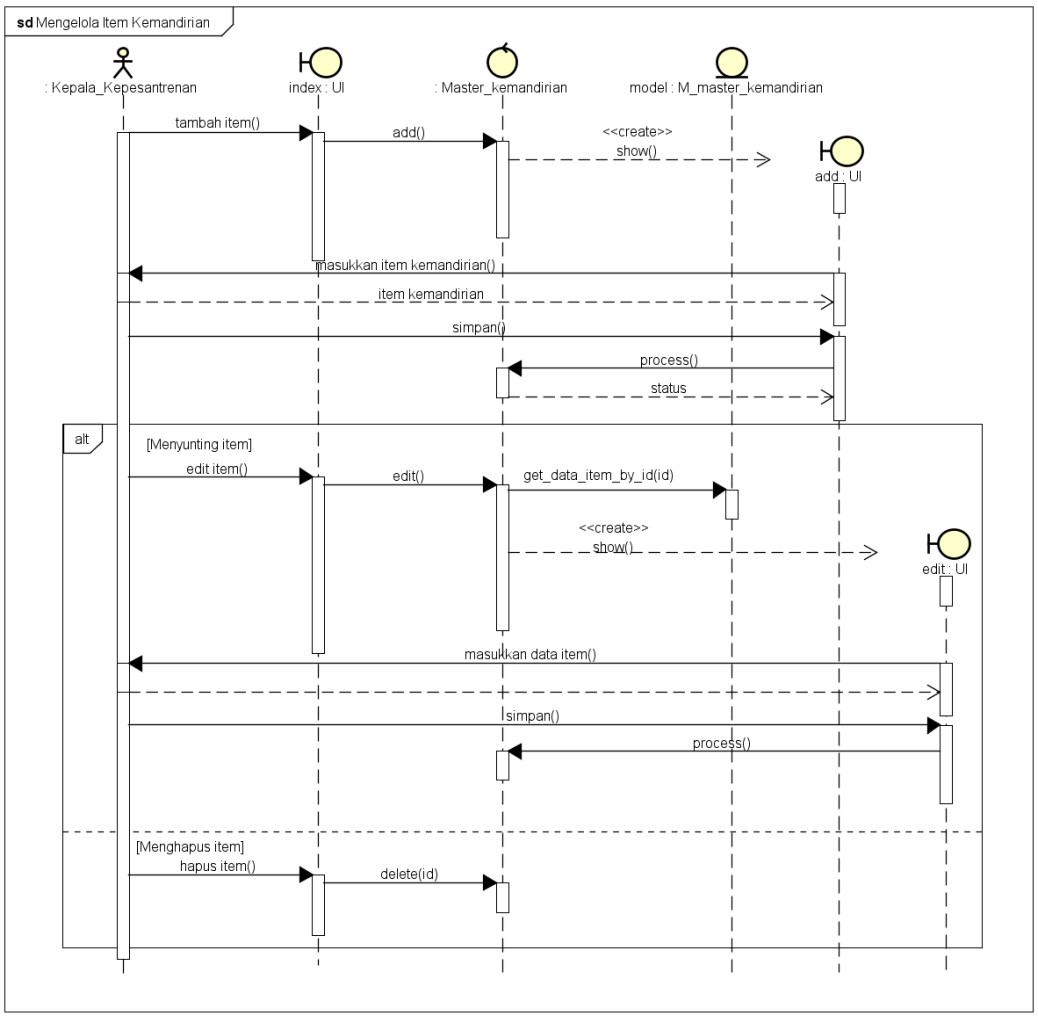
4. *Sequence Diagram* Mengelola Kamar Santri



5. *Sequence Diagram* Mengelola Murabbi Kamar



6. *Sequence Diagram* Mengelola Item Kemandirian



7. *Sequence Diagram* Mengelola Item Peribadatan